Frequency Assignment

Models and Algorithms

Frequency Assignment

Models and Algorithms

Frequentie Toewijzing

Modellen en Algoritmen

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Universiteit Maastricht,
op gezag van de Rector Magnificus, Prof. dr. A.C. Nieuwenhuijzen Kruseman,
volgens het besluit van het College van Decanen, in het openbaar te verdedigen op
donderdag 4 november 1999 om 16.00 uur

door

Arie Marinus Catharinus Antonius Koster

Promotor:

Prof. dr. ir. A.W.J. Kolen

Co-promotor:

Dr. ir. C.P.M. van Hoesel

Beoordelingscommissie:

Prof. dr. H.J.M. Peters (voorzitter)
Prof. dr. J.K. Lenstra (Technische Universiteit Eindhoven)
Dr. F.C.R. Spieksma

Frequency Assignment - Models and Algorithms
© Arie M.C.A. Koster, 1999
Proefschrift Universiteit Maastricht
ISBN 90-9013119-1

Preface

Voorwoord

From April 1, 1996 men in the Netherlands are not obliged anymore to join the military service. However, in the spring of 1995, I received my summons to join the service starting March 1996. So, I would belong to the last of the Mohicans who has to perform his duty. However, at more or less the same moment, Antoon Kolen and Olaf Flippo offered me a possibility to escape. They asked me to become a Ph.D. student at Maastricht University for the next 4 years. So, I had to make a decision between 9 months of military service and 4 years Ph.D. studentship. About four and a half years later, this thesis clarifies my choice: I became a Ph.D. student.

The choice to accept a Ph.D. studentship was, of course, not only made by the alternative of serving the country as a soldier. The application of mathematics to real problems attracted my attention from the start of my studies in Technical Mathematics at Delft University of Technology in 1991. I specialized in operations research, and in the last year I met two other students, who did their graduation project on frequency assignment. Their subject called my attention to the operations research problems in telecommunication, which resulted in my application for the position in the project *Combinatorial Optimization problems in Telecommunication*, with an emphasis on the frequency assignment problem. Antoon and Olaf offered me the freedom to do research on network design problems as well. Altogether, an attractive position in which I could do research to solve real-life operations research problems.

After finishing my Master's Thesis and a holiday, I started in Maastricht in September 1995. Soon, I discovered that Antoon and Stan van Hoesel had made a bet about the results of my research project. My task was to find the optimal solution for 11 frequency assignment problems. In case the optimal solution for an instance was better than the best known value (derived by Antoon), Antoon should pay Stan a bottle of wine. In case the optimal solution was equal to the best known one, a bottle of wine should go the other way. By now, we can conclude that Stan lost 7 bottles of wine to Antoon, whereas the bet has not ended yet for the 4 other frequency assignment problems.

It will be clear that Antoon, Stan and Olaf played an important role during my years in Maastricht. I owe a lot to them for their support, cooperation and friendship. Without the stimulating environment they created, this thesis would probably never have been completed. I am also indebted many thanks to them for the freedom they gave me to do research on other topics in telecommunication as well. To work on two different

topics, frequency assignment and network design gave me the opportunity to extend my knowledge to the whole area of operations research and telecommunication. The research on telecommunication network design resulted in three articles [56, 86, 132] that are not part of this thesis (they appeared in the Ph.D. thesis of Robert van de Leensel [131]).

Speaking of Robert, a special word of thanks should be devoted to him. He was for almost 4 years my roommate and co-researcher. The afternoons that we locked the door, and discussed research topics together are worthwhile remembering. They were pleasant and productive at the same time. Many times, Robert acted as guinea-pig to test the validity of new ideas. Besides the research activities, I also indebted thanks to him for our discussions on varying topics, the joint travel experiences (especially in the U.S.A. and Israel), and last but not least his friendship. Besides the already mentioned people, I have to thank the other members of the Operations Research group in the years 1995-1999: Joris van de Klundert, Ron van der Wal, Rudolf Müller, Jos Sturm and Jan-Willem Goossens. They were always available to answer my questions. Of course, also many thanks to the other members of the Department of Quantitative Economics for the pleasant contacts.

Furthermore, I have to thank the students in Econometrics for their interest in my research project. For four years I had the opportunity to teach them the basics and more advanced levels of programming in C++. Without mentioning particular students, very pleasant contacts originate from these courses. I also have to acknowledge some people from outside Maastricht. A couple of people served as hosts during the trip with Robert through the United States of America. The (financial) support of Martin Savelsbergh (Georgia Institute of Technology), Rutgers University, Oktay Günlük (AT&T Labs), Daniel Bienstock (Columbia University), David Williamson (IBM T.J. Watson Research Center), and 'Raghu' Raghavan (US West Telecommunications) provided that our trip ended successfully. The visit of the INFORMS Israel conference was financially supported by Shell Nederland.

Tot slot gaat mijn dank uit naar mijn familie en vrienden in Schoonhoven en omgeving. In het bijzonder moet ik mijn ouders en mijn zus Cora bedanken voor hun niet-aflatende steun en belangstelling voor mijn onderzoek en onderwijs. Veel dank ben ik verschuldigd voor hun ondersteunende activiteiten, zoals het vele wassen en strijken als ik weer eens met een uitpuilende tas met wasgoed thuiskwam. In de toekomst hoop ik jullie daar niet al te vaak meer mee lastig te vallen.

Finally, my latest but not least word of thanks has to go to our Creator, Who among all other things He did for me, talented me with the gift for mathematics. Without His never-ending aid and assistance, I would not have been able to complete this thesis.

Arie Koster
September 1999

Contents

Inhoudsopgave

Preface i

Contents ii

List of Figures v

List of Tables vii

1 Introduction 1
- 1.1 History of Wireless Communication . . . 1
- 1.2 Applications of Wireless Communication . . . 3
  - 1.2.1 Radio and television broadcasting . . . 3
  - 1.2.2 Terrestrial mobile cellular networks . . . 3
  - 1.2.3 Satellite-based cellular systems . . . 6
  - 1.2.4 Fixed cellular telecommunication networks . . . 6
- 1.3 Outline of this thesis . . . 7

2 The Frequency Assignment Problem: A Survey 11
- 2.1 The Frequency Assignment Problem . . . 12
- 2.2 Fixed Channel Assignment . . . 15
  - 2.2.1 Minimization of the Maximum Penalty . . . 15
  - 2.2.2 Minimization of the Cumulative Penalty . . . 16
  - 2.2.3 Frequency Assignment and Graph Coloring . . . 17
  - 2.2.4 Application specific properties . . . 18
- 2.3 Minimum Order Frequency Assignment . . . 20
  - 2.3.1 Problem Definition . . . 20
  - 2.3.2 Benchmark Instances . . . 22
  - 2.3.3 Lower Bounds and Exact Methods . . . 23
  - 2.3.4 Heuristics . . . 23
  - 2.3.5 Conclusions . . . 25
- 2.4 Minimum Span Frequency Assignment . . . 25
  - 2.4.1 Problem Definition . . . 25
  - 2.4.2 Benchmark Instances . . . 28
  - 2.4.3 Lower Bounds and Exact Methods . . . 29
  - 2.4.4 Heuristics . . . 32
  - 2.4.5 Conclusions . . . 33

2.5 Minimum Blocking Frequency Assignment . . . 34
2.5.1 Problem Definition . . . 34
2.5.2 Lower Bounds and Exact Methods . . . 36
2.5.3 Heuristics . . . 37
2.5.4 Conclusions . . . 38
2.6 Minimum Interference Frequency Assignment . . . 38
2.6.1 Problem Definition . . . 38
2.6.2 Benchmark Instances . . . 40
2.6.3 Lower Bounds and Exact Methods . . . 41
2.6.4 Heuristics . . . 41
2.6.5 Conclusions . . . 45
2.7 Other Models . . . 45
2.8 Conclusions . . . 46

3 The Partial Constraint Satisfaction Formulation 49
3.1 The partial constraint satisfaction problem . . . 50
3.2 Formulation, Dimension and Trivial Facets . . . 52
3.3 Lifting theorems . . . 54
3.4 Non-trivial classes of facets . . . 58
3.4.1 The cycle inequalities . . . 58
3.4.2 The clique-cycle inequalities . . . 60
3.5 Separation of non-trivial facets . . . 63
3.5.1 The cycle inequalities . . . 64
3.5.2 The clique-cycle inequalities . . . 70
3.6 The Boolean Quadric Polytope and the PCSP . . . 73
3.7 Computational Results . . . 77
3.8 Concluding Remarks . . . 80

4 A Tree Decomposition Approach 83
4.1 Graph Theoretic Concepts . . . 84
4.2 Construction of a Tree-Decomposition . . . 87
4.2.1 Minimum separating vertex set in a graph . . . 87
4.2.2 Heuristic . . . 88
4.3 Dynamic Programming Algorithm . . . 89
4.4 Reduction Techniques . . . 91
4.4.1 Constraint graph reduction . . . 92
4.4.2 Penalty shifting - Lower bounding . . . 94
4.4.3 Domain reduction . . . 94
4.5 Iterative Version Algorithm . . . 100
4.6 Computational Results . . . 104
4.6.1 Preprocessing . . . 105
4.6.2 Construction of Tree-decompositions . . . 106

4.6.3 Dynamic Programming Algorithm . . . 107
4.6.4 Iterative version . . . 109
4.6.5 Iterative Algorithm and Integer Programming . . . 111
4.7 Concluding Remarks . . . 112

5 Local Search Approaches 117
5.1 Preliminaries . . . 117
5.2 Local Search and Integer Programming . . . 118
5.2.1 Neighborhood . . . 119
5.2.2 Computational Results . . . 119
5.3 Local Search and Tree Decomposition . . . 121
5.3.1 Neighborhood . . . 122
5.3.2 Computational Results . . . 123
5.4 Concluding Remarks . . . 123

6 Directions for Further Research and Concluding Remarks 125
6.1 Directions for Further Research . . . 125
6.1.1 Benders Decomposition . . . 125
6.1.2 Lagrangean Relaxation . . . 126
6.1.3 Semi-Definite Programming Relaxation . . . 127
6.1.4 Frequency Assignment Formulation . . . 129
6.2 Conclusions . . . 132

Bibliography 135

Author index 149

Samenvatting in het Nederlands - Summary in Dutch 153
Introductie . . . 153
Frequentie Toewijzing: Een Overzicht . . . 154
Exacte Methoden voor het MI-FAP . . . 157
Heuristieken voor MI-FAP . . . 158
Richtingen voor Nader Onderzoek en Conclusies . . . 158

Curriculum Vitae 161

List of Figures

Lijst van Illustraties

1.1 Radio spectrum for wireless communication . . . 2
1.2 Subscribers GSM Networks . . . 4

2.1 Philadelphia instances. . . . 28
2.2 Reuse distances . . . 29

3.1 Extension of the graph . . . 55
3.2 Extension of the domain . . . 56
3.3 Cycle Inequalities . . . 59
3.4 Clique-Cycle Inequalities . . . 60
3.5 (γ, k)-Clique-Cycle Inequalities . . . 61
3.6 Digraph for the separation of cycle inequalities . . . 69

4.1 Example path decomposition . . . 85
4.2 Example tree decomposition . . . 86
4.3 Improvement step of a tree decomposition . . . 89
4.4 Heuristic for construction of a tree decomposition . . . 90
4.5 Dynamic programming algorithm . . . 92
4.6 Example shifting penalties . . . 94
4.7 Example upper bounding and dominance . . . 95
4.8 Example iterative algorithm . . . 101
4.9 Iterative version of the algorithm . . . 103
4.10 Example partition of subsets . . . 103
4.11 Graphical representation `CELAR 06` . . . 108
4.12 Number of non-redundant assignments `CELAR 06` . . . 110
4.13 Lower bounds `CELAR 06` . . . 112
4.14 Vertices and domain-subsets chart iterative algorithm . . . 113
4.15 Max number of non-redundant assignments iterative algorithm . . . 114

6.1 Example semi-definite relaxation . . . 129

List of Tables

Lijst van Tabellen

2.1 Minimum Order benchmark instances CALMA project 24
2.2 Characteristics Philadelphia benchmark instances. 30
2.3 Results Philadelphia benchmark instances 30
2.4 Minimum Span benchmark instances CALMA project 31
2.5 Minimum Interference benchmark instances CALMA project 42

3.1 Computational results $|D_v| = 2$ 78
3.2 Comparison exact and heuristic separation 78
3.3 Separation valid inequalities 79
3.4 Results on a subgraph problem with $|D_v| = 44$. 80

4.1 Penalties example upper bounding and dominance 96
4.2 Statistics and preprocessing CALMA-instances 106
4.3 Construction of a tree decomposition 107
4.4 Computational results dynamic programming algorithm test instances . . 109
4.5 Computational results dynamic programming algorithm 109
4.6 Computational results iterative version of the algorithm 111
4.7 Computational results iterative algorithm and integer programming 115

5.1 Results local search and integer programming 120
5.2 Local search and tree decomposition 124

6.1 Comparison integer programming formulations 131
6.2 Minimum Interference lower and upper bounds CALMA instances 133

1. Introduction

Mathematical models and algorithms for frequency assignment problems are the topic of this thesis. Frequency assignment problems occur in many different types of wireless communication networks. In the last decade, the rapid development of new wireless services like digital cellular phone networks resulted in a run out of the most important resource, frequencies in the radio spectrum. Like with all scarcely available resources, the cost of frequency-use provides the need for economic use of the available frequencies. Reuse of frequencies within a wireless communication network can offer considerable economies. However, reuse of frequencies also leads to loss of quality of communication links. The use of (almost) the same frequency for multiple wireless connections can cause an interference between the signals that is unacceptable. The frequency assignment problem balances the economies of reuse of frequencies and the loss of quality in the network. Quantification of the different aspects results in a mathematical optimization problem that can be solved with Operations Research techniques. Depending on the point of view of the researchers, the goal of the network provider, and application specific conditions many different models and algorithms are proposed.

The Chapters 2-6 are devoted to these models and algorithms. In this chapter we introduce the frequency assignment problem through a brief description of the early history of wireless communication networks in Section 1.1. In Section 1.2 we continue with the discussion of a number of applications of wireless communication in which the assignment of frequencies plays a crucial role. We conclude this chapter with an outline of the sequel of this thesis in Section 1.3. The contents of the Sections 1.1 and 1.2 is partly based on the encyclopedias [29] and [52].

1.1 History of Wireless Communication

More than a century ago, in the early 1890s, Marconi started to experiment with wireless communication via radio waves. His research resulted in 1897 in the successful transmission of radio waves to a ship at sea over a distance of 29 kilometers. A couple of years later, it was possible to transmit signals across the Atlantic Ocean, and already in 1905 many ships were using wireless telegraphy to communicate with shore stations. The importance of Marconi's invention can be demonstrated with for example the sinking of the Titanic. Without the equipment to communicate with other ships and shore stations, the

disaster would be considerably larger than it already was. In 1909, Marconi received the Nobel Prize in physics for his pioneering work on the wireless telegraph. Continuing improvements of the equipment resulted in the establishment of wireless telephony between Virginia and Paris in 1915. After World War I, radio broadcasting became more and more popular, first on an amateur level, later by professional broadcasters. Experimental television broadcasting already began in the 1930s and was successfully introduced to the mass since the end of the 1940s. In the last 50 years, the radio spectrum has been explored for wireless communication in many different ways. For example, space missions are not possible without communication via radio waves. It is not only used for voice communication with the astronauts, but also for the navigation of the spacecrafts. Nowadays radio waves are used for wireless telegraphy, radio broadcasting, television, cellular telephone networks, radar, navigational systems (air and sea traffic control), military communication, and space communication.

Every application uses a specific part of the radio spectrum. The frequencies that can be used for wireless communication range from 3 kilohertz to 300 gigahertz. These values correspond with a wavelength between 1 mm and 100 km. Figure 1.1 shows an overview of which frequencies are used for the different applications. The most popular applications, radio, television, and cellular phone use frequencies in the very high frequency (VHF) and ultra high frequency (UHF) spectrum. The use of frequencies for an application is regulated by the International Telecommunication Union (ITU) and national agencies. They issue licenses to use certain frequencies.

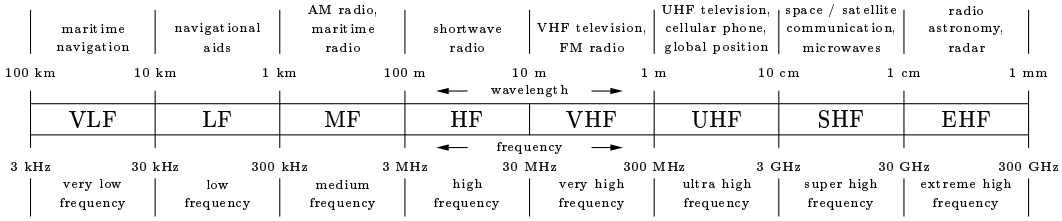


Figure 1.1: The radio spectrum that can be used for wireless communication.

Wireless communication between two points is established with the use of a transmitter and a receiver. The transmitter generates electrical oscillations at a radio frequency; the carrier frequency. Oscillations at a radio frequency can be modulated either via the amplitude or the frequency itself. The receiver detects these oscillations and transforms them in either sounds or images. When two transmitters use (almost) the same carrier frequency, they may interfere. The level of interference depends on many aspects like the distance between the transmitters / receivers, the geographical position of the transmitters, the power of the signal, the direction in which the signal is transmitted, and the weather conditions. In case the level of interference is high, the received signal may drop below the signal-to-noise ratio, which causes an unacceptable loss of quality. However,

the limited availability of frequencies causes their reuse by multiple transmitters within one and the same network.

As a consequence, an operator should carefully choose the frequencies on which each station transmits to avoid high interference-levels. The selection of the frequencies in such a way that interference is avoided, or second best, is minimized, is called the Frequency Assignment Problem (FAP). Depending on the application the conditions that should be satisfied by the frequency plan may vary. Therefore, it is not surprising that many different approaches have been suggested in the literature to solve this problem. In Chapter 2 we survey the most recent approaches. For now, we would like to illustrate the wide variety of FAPs with a brief discussion of the most popular applications. Successively, we discuss in the next section radio and television broadcasting, cellular telephone networks (both terrestrial and satellite based), and fixed location wireless telecommunication networks.

1.2 Applications of Wireless Communication

1.2.1 Radio and television broadcasting

Till the large-scale introduction of cable television and satellite broadcasting in the beginning of the 1980s, the most convenient way to transmit radio and television signals was (and still is for radio) through the air. To cover an area for radio or television broadcasting, antennae scattered around the region are necessary. For radio, each of these antennae transmit in a radial way either an Amplitude Modulated (AM) or a Frequency Modulated (FM) signal. For AM signals frequencies in the range from 540 kHz to 1600 kHz (medium frequency spectrum) are used, whereas FM signals are transmitted on VHF frequencies in the range 87 MHz - 108 MHz. Another part of the VHF spectrum and UHF frequencies are explored to transmit television signals. Like radio, the television signal is not directed but transmitted in a radial way. The signal transmitted by an antenna may not interfere with other signals transmitted by other radio or television services in the same area. Moreover, in case two antennae of the same broadcasting service cover a part of the area simultaneously, the signals of these antennae are not allowed to interfere as well.

1.2.2 Terrestrial mobile cellular networks

The last decade the development of (digital) cellular phone networks not only attracts the attention of the scientific community, but influences the whole society. Cellular phone networks, however, exist already more than 50 years. As early as 1946, the first commercial mobile telephone service (MTS) was introduced by AT&T. Eighteen years later, AT&T

introduced an improved version (IMTS). In this system 11 channels were available for all users within a geographic area. Since each frequency could be used only once (without interference), the IMTS system had a very limited capacity. In the region New York for example, the number of subscribers was limited to 545. In the late 1970s AT&T and Motorola Inc. developed the advanced mobile phone system (AMPS). AMPS had at its disposal 666 paired voice channels. These 666 channels made it possible to serve a large population. The system, publicly introduced in 1983, had 200,000 subscribers after the first year and 2,000,000 five years later. To increase the capacity even further, 166 additional channels were made available for the AMPS. Systems like the AMPS were also developed in Japan (1979) and Europe (starting in Scandinavia in 1981).

A disadvantage of the systems available in Europe during the 1980s was the incompatibility with one another. Therefore, in 1988 the Groupe Speciale Mobile (GSM) was founded by a group of government-owned telephone companies. They developed a new digital-based mobile communication standard, which is in use since 1992. GSM networks use frequencies in the 900 Mhz and 1800 Mhz spectrum (UHF). The GSM system became an overwhelming success, not only in Europe, but currently all around the world. At the end of 1998 a total of 320 networks in 118 countries were serving approximately 135 million customers (cf. Figure 1.2) [51, 77]. In the Netherlands, currently 5 providers of cellular

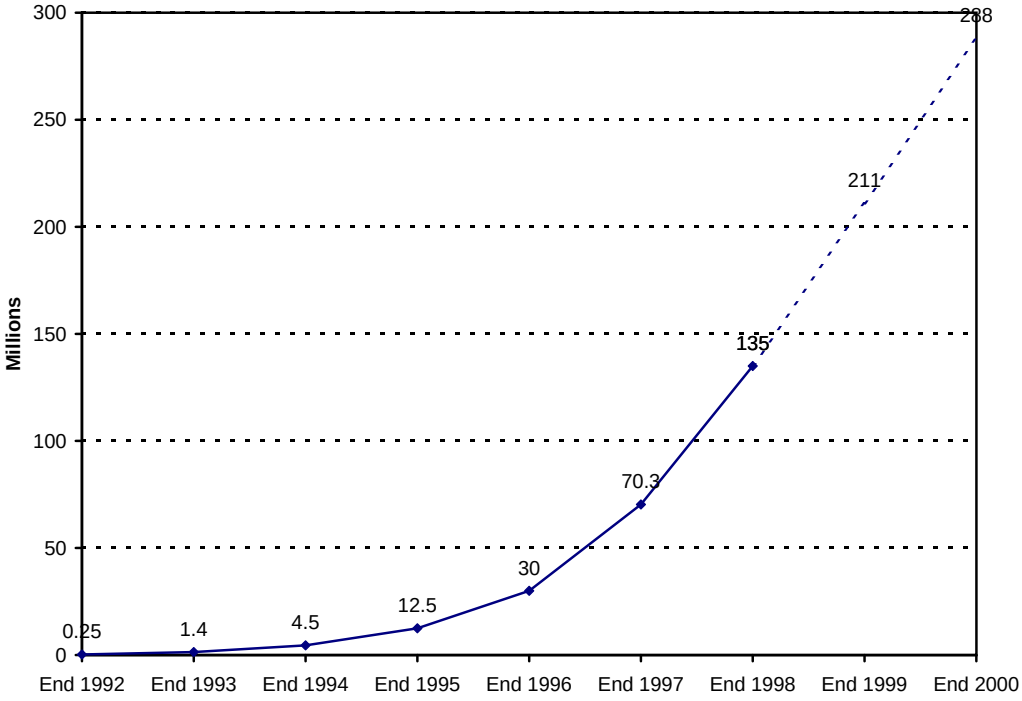


Figure 1.2: Number of subscribers to a GSM network [51, 77].

networks are active, serving more than 5 million customers (over 30% of the Dutch population) by the end of 1999 [151]. KPN Telecom, the former state telephone company,

started already in 1992 with its GSM network. In 1995 the new firm Libertel became the second provider of GSM in the 900 MHz spectrum in the Netherlands. Within 4 years Libertel became a company with more than 1.4 million customers, and a market value of 12 billion guilders[1]. After an auction of the available frequencies in the 1800 MHz spectrum in 1998, Telfort, Dutchtone and Ben started the construction of their cellular phone network.

A third generation of mobile communication will be introduced in the near future. UMTS (Universal Mobile Telecommunications System) will replace GSM as the new world wide standard for cellular phone networks. In contrast, with the current GSM system, UMTS will be compatible in both Europe and the United States of America. UMTS will use high speed connections (up to 2 megabit per second), which enables mobile use of internet and video communication.

All terrestrial cellular phone systems can be characterized by the following properties. They consist of a number of base stations that divide a geographic area to be served in smaller areas, called cells. Each base station operates on a certain frequency. A cellular phone within a cell is connected with the base station upon request via this frequency. As a mobile phone proceeds from one cell to another during a call, a mobile telephone switching office arranges that the call continues without noticeable interruption. If the demand for the wireless service within a cell exceeds the capacity of the base station, splitting the cell into smaller cells can increase the capacity. Depending on the geographical position, the power of the signal and the direction in which the signal is transmitted, transceivers may interfere when they use (almost) the same frequency.

The rapid development of cellular telephone networks in recent years has increased the need for good solution techniques for the frequency assignment problem for cellular networks. A main difference between radio / television broadcasting and cellular phone networks is the need for an individual connection for every customer. In radio and television networks thousands or even millions of customers receive the same signal transmitted by one single antenna. In a cellular phone network, a signal is transmitted only between one transmitter and one receiver. Technological developments like Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA), and Code Division Multiple Access (CDMA) make it possible to use a frequency simultaneously for a limited number of different customers within the same cell. Nevertheless, the service area of a cellular network has to be covered with a substantially larger number of antennae to satisfy the demand. In a country like the Netherlands for example, a radio or television network can cover the area with less than a dozen transmitters. In contrast, the number of transmitters needed to cover the same area for a mobile phone service can be as large as a couple of thousands. Combined with about 40 available frequencies, we have to conclude that each frequency must be reused many times. Fortunately, the low transmitting

[1] 1 guilder = 0.45378 euro

power of battery-operated portable phones offers the opportunity to increase the reuse of frequencies within the same service area without interference.

Not only, the scale of the cellular radio networks, but also the investments related to it, strengthen the need for good frequency plans. Let us illustrate the importance of a good assignment by the investments related to GSM networks in the Netherlands. The three providers Telfort, Dutchtone, and Ben paid altogether reputedly 1.8 billion guilders for the frequencies in the 1800 Mhz spectrum that were sold at an auction in 1998. Besides this investment, the building of a cellular network costs at least 300 million guilders. However, all these efforts are worthless without a good frequency plan that can guarantee high-quality communication links to the customers.

1.2.3 Satellite-based cellular systems

In addition to the terrestrial systems, in recent years satellite-based telephone systems are under construction. With such a system it is possible to make a connection with a cellular phone all around the world, especially in those regions currently not covered by a terrestrial system. In contrast with other satellite communication systems, satellite-based phone systems operate in the low earth orbit (780 km high), and are therefore called LEO systems. The satellites work differently from those at a much higher orbit (36,000 km) in two major ways. First, the small distance between earth and satellite makes it possible to connect to the system with a handheld device. Second, signals can be moved overhead in between satellites without the use of base stations at the earth [93]. The first LEO system is operated by Iridium Inc. [93], a consortium of corporations and governments from around the world. It consists of 66 satellites, forming a cross-linked grid above the Earth. In the near future, operators will supply services in which terrestrial and satellite communication are integrated.

Like in the terrestrial systems, the satellites operate on certain frequencies, that have to be selected in such a way that interference is avoided. Not only frequencies are needed to communicate with handheld devices, but also to establish communication between satellites and ground stations, and between satellites mutually.

1.2.4 Fixed cellular telecommunication networks

One of the most recent applications of wireless communication is the establishment of fixed cellular telecommunication networks. In contrast with mobile cellular networks, in non-mobile or fixed systems both the transmitters and receivers are located at fixed points in the area. Fixed cellular networks provide a financially attractive alternative to the construction of conventional wired networks in developing countries, where no wired

network structure is available yet. Moreover, the introduction of new services, like data communication (internet, e-mail) and video-conferencing causes shortage of capacity in existing wired networks. Point-to-point wireless connections can be used as an alternative to the extension of the capacity of these wired networks. In both cases no cable connections have to be established. A disadvantage of point-to-point connections is that the transmitter and receiver have *to see* each other, which means that there should be no obstacles in between them. As a consequence, transmitters and receivers have to be built at high locations (e.g., at the roof of apartment and office buildings). Although the transmitters are directed to the receivers their signals can interfere. Especially if signals cross each other, the use of (almost) the same frequencies should be avoided.

Another application that has similarities with fixed cellular networks stems from the military. In military communication networks, wireless connections have to be established between pairs of transceivers. These connections, or radio links, can interfere with each other, if they use similar frequencies in one and the same area.

1.3 Outline of this thesis

In the previous section we have discussed several applications of wireless communication. For each of these applications the quality of the network depends on a 'good' frequency assignment plan. The sequel of thesis is devoted to models and algorithms to solve the corresponding frequency assignment problems (FAPs). It can be divided in three parts. We start in **Chapter 2** with a survey on the models and algorithms that have been proposed for the FAP in the literature in recent years. We discuss four different ways to model the FAP. Each of these models has its own (dis)advantages. For every model, we investigate which approaches have been proposed in the literature. These approaches can be divided in three categories: (i) heuristic methods, like sequential assignment algorithms, Simulated Annealing, and Tabu Search, (ii) exact methods, like exhaustive search and integer programming techniques, and (iii) Lower bounds obtained from graph theory.

In the sequel of the thesis, we focus on one of these models, the minimum interference frequency assignment problem (MI-FAP). Many heuristic techniques have been applied to this problem. Exact methods and lower bounds, however, are rarely explored. Only very special cases can be solved with the existing exact methods. Therefore, the second part of the thesis is devoted to solution techniques for this fairly general model. Two exact methods are described in the Chapters 3 and 4, whereas Chapter 5 is devoted to new heuristics that are derived from the exact methods. In **Chapter 3**, we model the MI-FAP as a Partial Constraint Satisfaction Problem (PCSP). We present an integer linear programming formulation for the PCSP and study the corresponding polytope from a polyhedral point of view. We prove two lifting theorems for facet defining valid

inequalities. With these theorems we can analytically derive classes of facets, given a single facet for a particular problem. In fact, we derive two classes of facet defining valid inequalities in this way. For these two classes we discuss the complexity of the corresponding separation problems. Due to the equivalence of a restricted version of the PCSP and the boolean quadric polytope (BQP), other classes of facets can be obtained from facets for the BQP. The chapter is concluded with computational results on a number of instances. They show that the two classes of valid inequalities are very effective for MI-FAPs with small domains. However, for real-life instances, the inequalities, although they are facets of the polytope, are not powerful enough to close the gap between the linear programming relaxation and the optimal solution in reasonable time.

Due to the limited applicability of the integer linear programming techniques on real-life instances, we present in **Chapter 4** an approach based on the tree decomposition of the constraint graph. The MI-FAP is defined on a graph, and like many other combinatorial optimization problems based on graphs, it can be solved in time polynomial in the length of the input, in case the treewidth of the graph is bounded by a constant. Although this fact is well known, computational studies in this direction have not been described in the literature for any combinatorial optimization problem. The only study we are aware of has been carried out by Cook in the context of the traveling salesman problem [37]. In Chapter 4, we study the use of tree decompositions of the constraint graph to solve the FAP. In fact, we describe a dynamic programming algorithm to solve the FAP to optimality in polynomial time, given that the treewidth of the constraint graph is bounded by a constant. The algorithm, however, is exponential in the treewidth, which causes the need for additional processing techniques to reduce the time and memory efforts of the algorithm. We describe methods to reduce the size of the graph, as well as methods to reduce the size of the domains and the number of different assignments we have to memorize. Computational experiments show that small-sized and medium-sized real-life instances can be solved using these additional (pre)processing techniques. For the more difficult instances, however, the method still requests too much time and memory. Therefore, we present an iterative version of the algorithm which can be used to obtain a sequence of non-decreasing lower bounds for the original problem. For the more difficult instances we can derive the first non-trivial lower bounds in this way. Combination of the iterative techniques of Chapter 3 resulted in even better lower bounds for the instances with large treewidth.

We conclude the second part of the thesis with the discussion of two new heuristics for the MI-FAP. In **Chapter 5** we present two local search algorithms based on the results of Chapters 3 and 4. In the first local search algorithm the question whether there exists a neighbor in the solution space with smaller objective value is answered by solving a PCSP with 2 domain elements. In Chapter 3 it is shown that this problem, although $\mathcal{NP}$-complete can be solved efficiently by the polyhedral results. The second local search approach is based on the fact that PCSP on graphs with small treewidth can be solved with

dynamic programming. This result can be incorporated within a local search framework. Instead of solving the PCSP on the complete graph, we solve subproblems on induced subgraphs with small treewidth. Preliminary computational results show that both local search algorithms produce promising results.

In the last chapter of this thesis, **Chapter 6**, we address directions for further research on the MI-FAP, and state overall concluding remarks. We briefly discuss other exact methods like well known techniques as Benders Decomposition, and Lagrangian relaxation. We also discuss a semi-definite programming relaxation. All these methods and formulations can be used to solve the more general PCSP, instead of the MI-FAP. The last direction for further research is dedicated to a new integer linear programming formulations for the MI-FAP, in which the number of variables is substantially smaller than in the PCSP formulation of Chapter 3. This formulation is based on characteristics typical for a MI-FAP, and therefore cannot be used to solve the more general PCSP. The thesis is closed with some conclusions concerning the results of this thesis.

2. The Frequency Assignment Problem: A Survey

Frequency Assignment Problems (FAPs) have been investigated by many researchers. Depending on the application and the goal of the researchers, a wide variety of models and solution techniques have been proposed. The use of a wide variety of methods is prompted by the fact that the FAP belongs to the class of $\mathcal{NP}$-complete problems, which means that there does not exist an algorithm that solves the problem in time polynomial in the length of the input, unless $\mathcal{P} = \mathcal{NP}$ (see either Garey and Johnson [62] or Papadimitriou and Steiglitz [153] for a discussion of $\mathcal{NP}$-completeness). In this chapter we present a survey on the models and algorithms that have been proposed in recent years. We start with a general description of the FAP in Section 2.1. We also discuss in this section the 3 approaches applied to the FAP: Fixed Channel Assignment schemes, Dynamic Channel Assignment schemes, and Hybrid Channel Assignment schemes. Our survey is devoted to fixed channel assignment schemes. In Section 2.2 we distinguish the 4 most accepted points of view for the FAP. For each of the models, we compare as far as possible the results obtained by the different techniques available to solve combinatorial optimization problems in the Sections 2.3-2.6. Section 2.7 is devoted to approaches which cannot be classified within one of the four models. We close this survey with some conclusions in Section 2.8. The survey in [4] will partly be based on this chapter. Recently, an overview of exact methods for frequency assignment is given by Jaumard, Marcotte and Meyer [100].

In the sequel of this chapter many different approaches from the fields of operations research and artificial intelligence are discussed. For those not familiar with one or more of these methods we refer to the following papers and books on the topic:

Local Search (general). For local search techniques in general we refer to the recent book of Aarts and Lenstra [8], which gives a comprehensive overview of the available techniques.

Tabu Search. Tabu search was introduced by Glover [68], and more recently discussed in [69, 70, 71] and the book by Glover and Laguna [73]. In Glover [72], tabu thresholding was introduced, a combination of tabu search and candidate list strategies.

Simulated Annealing. Kirkpatrick, Gelatt and Vecchi [115] introduced the use of simulated annealing to optimization problems. The book by Aarts and Korst [7] discusses the topic comprehensively.

Genetic Algorithms. Genetic algorithms have been proposed by Holland [87], and are discussed in Goldberg [74] as well.

Neural Networks. Neural networks were introduced in the field of optimization by Hopfield and Tank [88, 89] (see also Dayhoff [44]).

Integer Programming. For exact methods based on integer linear programming like Branch and Bound, Branch and Cut, and Column generation, we refer to Nemhauser and Wolsey [148] or Schrijver [166]. A comprehensive overview of network flow problems is presented by Ahuja, Magnanti, and Orlin [9].

2.1 The Frequency Assignment Problem

The frequency assignment problem has two basic aspects:

(i). a set of wireless communication connections must be assigned frequencies such that data transmission between the transmitter and receiver for every connection is possible. The frequencies should be selected from a given set that may differ among connections. Note that much traffic is bidirectional, so that in fact two frequencies must be chosen, one for each direction.

(ii). The frequencies assigned to two connections may incur interference resulting in loss of quality of the signal. Two conditions must be fulfilled in order to have interference of two signals:

  (a) The two frequencies must be close on the electromagnetic band (Doppler effects) or (close to) harmonics of one another. The latter effect seems to be limited, however, since the frequency bands from which we can choose are usually so small that they do not contain harmonics.

  (b) The connections must be geographically close to each other. The signals that may interfere should have a similar level of energy at the position where they might disturb each other.

Both aspects are modeled in many different ways in the literature. The models discussed in the literature differ in the types of constraints they impose on frequency choices for connections they make, and in the objectives to be optimized. We will describe the practical settings of known applications, and the simplifications that are assumed in the accompanying models that lead to the models described in the literature. The diverse models are discussed both in their common features and their differences.

The frequency band $[f_{min}, f_{max}]$ available to some provider of wireless communication is usually partitioned in a set of channels, all with the same bandwidth Δ of frequencies.

For this reason the channels are usually numbered from 1 to a given maximum N, where $N = (f_{max} - f_{min})/\Delta$. The available channels are denoted by the domain $D = \{1, \ldots, N\}$. For a particular connection possibly not all channels from D are available. For instance, if this connection is close to the border of a country division rules between the countries involved may lead to a substantial decrease in channel availability. Therefore, the channels available for a connection v form a subset $D_v \subseteq D$. On each channel available one can communicate information from a transmitter to a receiver. For bidirectional traffic one needs two such channels, one for each direction. In the models considered in the literature the second channel is almost always ignored. The reasons for ignoring this aspect of the FAP depend on the application. Instead of one band $[f_{min}, f_{max}]$, in most applications two bands $[f^1_{min}, f^1_{max}]$ and $[f^2_{min}, f^2_{max}]$ of N channels are available: one with the channels $\{1, \ldots, N\}$, and one with the channels $\{s + 1, \ldots, s + N\}$, where $s \gg N$. Thus, the backward connection uses a channel which is shifted s channels up. The choice of s prevents any interference of backward channels with forward channels. As a consequence, each assignment for the forward channels can directly be transformed to an assignment for the backward channels with similar performance.

Interference of signals is measured by the signal-to-noise ratio (or signal-to-interference ratio) at the receiving end of a connection. There, the signal transmitted should be clearly understandable. The noise comes from other signals which have an interfering frequency on which they broadcast. There may be more than one source that transmits on the same or a close frequency and thus contribute to the total noise experienced at the receiver. In practice, a threshold value of 12 dB or 15 dB for the signal-to-noise ratio is found satisfactory. The computation of the level of interference is a difficult job in itself, since it depends not only on signal choice and strength, but also on the shape of the environment. If we ignore the environment and consider some other signal transmitted at the same frequency channel, then the interference of this signal at the receiver is computed with the following formula: P/d^γ where P is the power of the transmitter and d the distance to the disturbed receiver. Here, γ is a fading factor with values between 2 and 4. If this other signal is transmitted on a frequency at a distance of $n \geq 1$ units of the original signal, then a filtering factor of $-15(1 + \log^2 n)$ is to be taken into account (see [48]). The fact that multiple signals may disturb communication quality is ignored in most models, where only interference between pairs of connections is measured. A notable exception is [54], in which constraints are developed to determine the total interference of neighboring connections. Another assumption is the quantification of the levels of interference. We will assume given values of the interference as input to our models and problems.

The previously discussed two-way traffic poses another problem, even in the binary case, since interference need not be symmetric: if a transceiver[1] pair (r_1, r_2) transmits on frequencies f and $f + s$, and another transceiver pair (s_1, s_2) transmits on frequencies g and $g+s$ where f and g interfere, and $f+s$ and $g+s$ interfere, interference levels at r_1 and

[1] A single unit containing a transmitter and a receiver

r_2 may be different due to the fact that these transceivers may have different distances to s_1 and s_2. As mentioned before, this aspect is ignored in most of the literature.

Depending on the application, one or multiple connections have to be established between the same geographic end points. In general, this is modeled by assuming that $c_v \in \mathbb{Z}^+$ frequencies have to be assigned to connection v. Interference between frequencies assigned to the same connection can be avoided by the introduction of an additional value for certain combinations of frequencies $f, g \in D_v$. In practice the values c_v vary during time, depending on the actual demand for connections. By this property, the approaches suggested in the literature to deal with the FAP can be divided into three categories: Fixed Channel Assignment (FCA), Dynamic Channel Assignment (DCA), and Hybrid Channel Assignment (HCA) schemes[2].

In an FCA the forecasted demand is transformed to the requirement that we have to assign to each connection a number of frequencies beforehand. In this scheme it is not allowed to change the assignment on-line to satisfy actual demand for wireless connections. This is in contrast with DCA schemes in which frequencies are assigned on-line to the connections in such a way that the actual demand is satisfied and the interference is minimized. An example of a procedure for DCA is presented by Janssen, Kilakos and Marcotte [98]. They discuss a fixed preference assignment scheme. For every cell there exists a preference list of frequencies to serve the demand. In [98] it is proved that the preference lists can be constructed in such a way, that they are optimal according to some performance measure.

Finally, in HCA schemes a combination of FCA and DCA is implemented to obtain a better overall performance of the network. In HCA schemes a number of frequencies is assigned to every connection beforehand, whereas another part of the spectrum can be used for on-line assignment of frequencies upon request. An example of an HCA scheme is given by Sandalidis, Stavroulakis and Rodriguez-Tellez [165], who describe a neural network and a genetic algorithm approach to the borrowing channel assignment problem. In the borrowing channel assignment problem, a fixed number of frequencies are assigned to the connections. However, whenever the actual demand for frequencies exceeds the number of available frequencies, the connection can borrow an unused frequency assigned to an adjacent connection. The performance of networks that operate on DCA and HCA schemes is mostly studied via simulation of the particular procedure. It is proved by Johri [104] that DCA schemes perform better than FCA schemes under light traffic and under nonuniform traffic. However, under uniform and heavy load, FCA schemes outperform the DCA schemes. Besides this, FCA gives a bound on the performance of the DCA scheme. In fact, in case the DCA scheme allows for complete rearrangement of the assignment, a FCA problem has to be solved, every time the situation changes. For these last two reasons, we concentrate in this thesis on fixed channel assignment.

[2]The need for multiple connections, and the application of DCA and HCA schemes originates from cellular phone networks.

For a survey on the topic of FCA, DCA and HCA schemes we refer to Katzela and Naghshineh [111].

2.2 Fixed Channel Assignment

In a Fixed Channel Assignment scheme, the expected traffic load of the network is transformed to a requirement that we have to assign to each connection a fixed number of frequencies. The standard representation of a FAP is by means of a graph $G = (V, E)$, the *interference graph* or *constraint graph*. Each connection is represented by a vertex $v \in V$. The available channels or frequencies for a vertex are denoted by the set $D_v \subseteq D$. Let c_v denote the required number of frequencies for connection $v \in V$. Two vertices v and w for which the corresponding connections may interfere for at least one pair of frequencies, are connected by an edge $\{v, w\} \in E$. For each pair of frequencies $f \in D_v$ and $g \in D_w$ we penalize the combined choice by a measure depending on the interference level. This penalty is denoted by p_{vwfg}. The interference between two frequencies $f, g \in D_v$ assigned to the same vertex v can be modeled in the same way: an edge $\{v, v\} \in E$ and penalty p_{vvfg}. Another way to model this, is by replacing v by c_v vertices and additional edges between all of them. Some instances deal with a frequency plan in which changes are considered, to reduce interference. This reduction should take place under minimal changes of the total frequency plan, thus changes in the plan are penalized per change as well. This is modeled with additional penalties on the frequencies to be chosen for each vertex: the choice of frequency $f \in D_v$ costs q_{vf}.

The approaches to solve the FAP can be subdivided in two main streams. We want to assign frequencies to the vertices in such a way that either the total penalty incurred by a solution (minsum) or maximum penalty incurred by a solution (minmax) is minimized.

2.2.1 Minimization of the Maximum Penalty

Instead of computing a solution where the maximum penalty is minimized, we often search for a solution where the incurred interference does not exceed a given threshold value. Thus, certain frequencies and combinations of frequencies are forbidden. This essentially reduces the penalty matrices of the edges to 0-1 matrices. Combinations of frequencies with penalty 0 are allowed, whereas penalty 1 is forbidden. The objective reduces in this case to find a feasible solution, i.e., a solution in which no forbidden combinations are selected. In case such an assignment exists, often a second objective is introduced. The second objective represents a preference relation between all feasible assignments. In the 1970s minimization of the number of used frequencies was a popular second objective, since frequencies should be bought per unit at a high price in that time. The problem

of minimizing the number of used frequencies is called the minimum order problem, or minimum cardinality problem. The objective to minimize the span, i.e., the difference between the highest and lowest frequencies selected, was popular in the 1980s where frequencies were bought per bandwidth, e.g., the value N determined the costs.

In case the incurred penalties exceed the given threshold in every assignment, two directions remain. On the one hand, the threshold value can be increased allowing for assignments with more interference. On the other hand, we can search for a partial assignment that does not exceed the given threshold penalty. The most common objective to distinguish between partial assignments is minimization of the blocking probability. In case instead of the requested c_v only m_v frequencies can be assigned to a connection, we can calculate the probability that in practice a request to establish a connection have to be rejected. This probability is called the blocking probability of the connection. Optimal partial assignments minimize the overall blocking probability of the network.

2.2.2 Minimization of the Cumulative Penalty

The minsum criterion is not seen frequently in practical instances, but in some models it is combined with the minmax criterion by introducing threshold values for the penalties that denote the maximum acceptable interference. We then look for feasible solutions with a minimum total penalty under the condition that no penalty exceeds the threshold value. This combined model is most accurate in describing real-world problems, but it is also the one for which it is most difficult to determine optimal solutions. Note that the combined model is easily translated into a minsum problem, by setting all penalties exceeding the threshold to infinity.

Summarized, we can distinguish four models to solve the FAP. In this chapter we discuss these four most common points of view:

(i). The Minimum Order Frequency Assignment Problem (MO-FAP),

(ii). the Minimum Span Frequency Assignment Problem (MS-FAP),

(iii). the Minimum Blocking Frequency Assignment Problem (MB-FAP), and

(iv). the Minimum (Total) Interference Frequency Assignment Problem (MI-FAP).

In the Sections 2.3-2.6, we discuss these models (and their variants) in more detail. In Section 2.7 we mention some examples of other fixed channel approaches to the FAP that cannot be classified in one of the above models. We close this section with a discussion of the relation of frequency assignment with graph coloring (Section 2.2.3), and a description of application specific properties (Section 2.2.4).

2.2.3 Frequency Assignment and Graph Coloring

The minmax criterion is closely related to generalized coloring problems like T-coloring or list coloring. This relation has first been observed by Metzger [145] (see also Hale [78]). The relation of finding a feasible solution with coloring is due to two modeling aspects. First, the division of the levels of interference in acceptable and unacceptable levels, reduces the problem to forbidden combinations and allowed combinations like in the graph coloring problem, where it is not allowed to color two adjacent vertices with the same color. Second, in many settings the interference levels are related with the distance between the assigned frequencies: the smaller the distance between two assigned frequencies, the larger the interference level p_{vwfg}. Combined with the first assumption, interference is defined unacceptable between the connections v and w if $f \in D_v$ and $g \in D_w$ are at a distance smaller than d_{vw} of each other. The interference is acceptable if the frequencies are at a larger distance, i.e., $|f - g| \geq d_{vw}$. Now, if we relax the problem by setting all $d_{vw} = 1$ for $\{v, w\} \in E$, which is not far beyond reality, then we only penalize equal choices of frequencies for connected vertices. Thus, we can view the frequencies as colors, and a solution should have as few (or none at all) edges for which the end vertices have the same colors.

In a more general setting, we are not allowed to assign frequencies that differ a value contained in a set T_{vw} (containing 0), i.e., $|f - g| \notin T_{vw}$. If the set T_{vw} is defined by $\{0, \ldots, d_{vw} - 1\}$ the problems are equivalent. However, more general sets with non-consecutive numbers may also be defined. For instance, in the context of UHF television broadcasting the set T_{vw} consists of a non-successive set of integers (see Hale [78]). In case all sets T_{vw} are the same, the problem reduces to a T-coloring problem, which was introduced by Hale [78]. He formally defined both the minimum order and minimum span variants of the T-coloring problem, and connected them to the frequency assignment problem.

Another way to represent the minimum distance constraints is the use of a compatibility matrix C where the rows and columns correspond to the connections. The values $C_{vw} := d_{vw}$ denote the minimum separation distance of the frequencies. In case $C_{vw} = 0$, the vertices v and w are not adjacent, and no constraint on the assigned frequencies exists. In the literature the constraints are called differently depending on the value C_{vw}. We distinguish *co-site*, *adjacent-site*, *co-channel*, and *adjacent-channel* constraints. The values C_{vv} are called the *co-site* constraints, whereas the values $C_{vw} > 0$ are the so-called *adjacent-site* constraints. The terms *co-channel* and *adjacent-channel* constraints are widely used to designate a difference between values $C_{vw} = 1$ (we are not allowed to assign the same frequencies to both connections), and values $C_{vw} \geq 2$ (we are not allowed to assign adjacent channels to the connections), respectively.

2.2.4 Application specific properties

To conclude this section we describe a number of application specific properties and instances that are discussed in the literature. The large variety in practical settings does not only lead to these different models for the FAP, but also to different instance types. Some of the settings are:

Mobile telephone. This application differs from the standard FAP in the sense that one of the endpoints of the connection is a fixed antenna, and the other endpoint is a mobile phone. Each antenna covers a certain area, where it can pick up signals from mobile phones. The frequency chosen for a certain connection is determined only by use of the position of the antenna, and by the positions of the neighboring antennae that cover part of the area simultaneously. Therefore, vertices in the constraint graph do not correspond to connections, but to antennae, i.e., only one of the two end vertices of a connection. Antennae are usually concentrated into cells. Each cell contains about 4 antennae. At a site there are several cells present, usually about 4 or 5. In the literature frequency restrictions are often presented by means of the so-called re-use distances. If an antenna uses a frequency f, then there is a large area around it with radius d^0 in which this frequency cannot be re-used by some other antenna. In general a re-use distance d^s is defined as the area around an antenna where no frequencies in the interval $[f - s, f + s]$ can be used by other antennae. In terms of the constraint graph this means that two antennae v and w within a geographical distance d^s of each other can not be assigned frequencies at a distance of s or smaller. Thus, there is an edge between v and w with a constraint defined by frequency distance s. In most practical applications the re-use distances d^0 and d^1 are applied. The first denotes the co-channel interference. It induces a lot of edges in the constraint graph with distance 1. The second is the adjacent channel interference. It induces usually edges between antennae of the same site or cell. Below we give a list of (real-life) instances known from the literature with their specifics.

- The Philadelphia instances were among the first discussed in the literature [12]. The sites in Philadelphia were modeled on a hexagonal grid (see Figure 2.1, page 28). Each site demands a high number of frequencies, the multiplicity of the sites. Adjacent sites should not be admitted to use the same frequencies, and antennae at the same site should not use adjacent frequencies either. Variants of this structure use different re-use distances, for instance $(d^0, d^1, d^2) = (3, 2, 1)$ meaning that antennae in cells of distance 3 can use the same frequency and so on (the distance between 2 adjacent cells is taken as unit distance). The Philadelphia instances were used in many studies to explore lower bounding techniques on the span of instances, see Section 2.4.
- Castelino, Hurley, and Stephens [33] discussed 6 computer generated realistic

instances that have comparatively high distances in the constraint graph, and are fairly large with respect to the number of antennae. For every antenna 50 frequencies are available. The objective is to minimize the interference (see Section 2.6).

- Hao, Dorne, and Galinier [81] (see also [45, 46, 47, 80]) used instances from the French National Research Center for Telecommunications (CNET). They searched for interference free assignments with a minimal number of frequencies. The number of frequencies needed varies from 8 to 30, whereas the number of vertices in the constraint graph is at most 300 ($c_v = 1$ or 2).
- Borndörfer et al. [23, 24] studied also cellular phone instances from one of the German telecommunication companies (E-Plus). The size of the instances ranges from 267 vertices (20,164 edges) to 4,240 vertices (529,000 edges). The number of available channels varies from 30 to 50. The objective is to minimize the cumulative interference.
- Finally, Crisan and Mühlenbein [40] carried out experiments on 6 real world examples with up to 5,500 vertices and $3.9 \cdot 10^5$ constraints. The objective is to minimize the total interference.

Military applications. In military the usage of field phones leads to (in principle dynamic in time and place) FAPs. These FAPs have the property that each connection consists of two movable phones. To each connection we have to assign two frequencies at a fixed distance of each other. Thus, all frequencies are given as combinations of two with this fixed distance. Instances are available in the EUCLID CALMA (Combinatorial ALgorithms for Military Applications) project. In the CALMA project researchers from England, France, and the Netherlands tested different combinatorial algorithms on the same set of frequency assignment problems. The set contains minimum order problems (cf. Section 2.3), as well as minimum span (cf. Section 2.4) and minimum interference problems (cf. Section 2.6). Eleven real-life instances were provided by CELAR (Centre d'ELectronique de l'ARmement, France), whereas a second set of 14 instances was made available by the research group of Delft University of Technology. These GRAPH (Generating Radio Link Frequency Assignment Problems Heuristically) instances were randomly generated by Van Benthem [18], and have the same characteristics as the CELAR instances. Besides the minimum distance constraints, the instances also contain equality constraints, to model that two frequencies at a fixed distance have to be assigned to the corresponding vertices. The distance is the same for all constraints and every vertex is contained in exactly one equality constraint. Moreover, the domains are constructed in such a way that for every frequency there exists only one 'matching' frequency. Altogether, these characteristics provide the possibility to reduce the size of the instances to half the original size whenever that may be profitable. The number of available frequencies for a vertex is 40 on average. We have to assign one frequency to every vertex ($c_v = 1$). Results of the CALMA project as well as

all test problems are available by anonymous ftp [32]. Aardal et al. [6] (extended abstract published as [5]) presented an overview of the different approaches applied in connection with the project (see also [183, 31]).

Radio and Television. These are applications where the standard FAP arises in a special setting with regard to the instances, but essentially these instances resemble the mobile phone instances. In [140] instances of this type, provided by a large Italian broadcasting company are discussed.

Satellite communication. In Thuve [181] a frequency planning problem in satellite communication is discussed. In this application both the transmitters and receivers are ground terminals. They communicate with each other with the help of one or more satellites. Each signal is first transmitted via an uplink to the satellite and next transmitted by the satellite via a downlink to the receiving terminal. The uplink and downlink frequency should be separated by a fixed distance. This distance, however, is larger than the bandwidth, which implies that we only have to assign frequencies to the uplink. Instead of assigning c_v frequencies to connection v, we have to assign N_l consecutive frequencies in this problem. To avoid interference, every frequency may be used only once. The objective is minimization of the cumulative interference related to the single assignments.

2.3 Minimum Order Frequency Assignment

2.3.1 Problem Definition

In the minimum order frequency assignment problem (MO-FAP), we have to assign frequencies in such a way that no unacceptable interference occurs, and the number of different used frequencies is minimized. Formally we can describe the problem as follows:

Minimum Order Frequency Assignment (MO-FAP)
INSTANCE: Undirected graph $G = (V, E)$, $\{v, v\} \in E$, for all $v \in V$, sets $T_{vw} \subset \mathbb{Z}$, $\{v, w\} \in E$, $0 \in T_{vw}$, demand $c_v \in \mathbb{Z}^+$, domain subsets $D_v \subseteq \mathbb{Z}^+$ for all $v \in V$, $D = \cup_{v \in V} D_v$, and positive integer K.
QUESTION: Does there exist an assignment of subsets $f : V \mapsto 2^D$ such that,

(i). $|f(v)| = c_v$,

(ii). $f(v) \subseteq D_v$,

(iii). $|\bar{f} - \bar{g}| \notin T_{vw}$ for all $\{v, w\} \in E$, $\bar{f} \in f(v)$, $\bar{g} \in f(w)$, $v \neq w$ or $\bar{f} \neq \bar{g}$, and

(iv). $|\cup_{v \in V} f(v)| \leq K$?

The MO-FAP is the first frequency assignment problem that is discussed in the literature. In most articles, Metzger [145] has received the credits for bringing the MO-FAP to the attention of the Operations Research society (cf. Hale [78]). This problem is a direct generalization of the graph coloring problem.

Graph K-Colorability (Garey and Johnson [62])
INSTANCE: Undirected graph $G = (V, E)$, and positive integer $K \leq |V|$.
QUESTION: Is G K-colorable, i.e., does there exist a function $f : V \mapsto \{1, 2, \ldots, K\}$ such that $f_v \neq f_w$ whenever $\{v, w\} \in E$?

The minimum number of colors needed to color the graph is denoted as $\chi(G)$. Karp [110] proved that Graph K-Colorability is $\mathcal{NP}$-complete for all $K \geq 3$. As a consequence MO-FAP is $\mathcal{NP}$-complete as well. In Garey and Johnson [61], it is proved that approximation of the optimal value within a factor 2 is $\mathcal{NP}$-complete as well. A generalization of graph coloring (and restricted version of MO-FAP) is proposed in Hale [78], and is well known as T-coloring.

Minimum Order T-Coloring (Hale [78])
INSTANCE: Undirected graph $G = (V, E)$, set $T \subset \mathbb{Z}^+$, $\{v, w\} \in E$, $0 \in T$, and positive integer K.
QUESTION: Does there exist an assignment $f : V \mapsto \mathbb{Z}^+$ such that, $|f_v - f_w| \notin T$ for all $\{v, w\} \in E$ and $|\cup_{v \in V} f_v| \leq K$?

The minimum number of colors needed to color a graph G with respect to the set T is denoted by $\chi_T(G)$. Cozzens and Roberts [39] proved that $\chi_T(G) = \chi(G)$: Let $(f_v)_{v \in V}$ be a coloring for G, and let $t_{\max} = \max_{t \in T} t + 1$. Then the coloring $(t_{\max} f_v)_{v \in V}$ is a feasible T-coloring.

As a consequence, research in this direction has been focused on the graph coloring problem instead of the T-coloring problem, or on the minimum span T-coloring problem (cf. Section 2.4).

Another generalization of graph coloring (and restricted version of MO-FAP) is the List Coloring problem.

Minimum Order List Coloring (Erdös, Rubin, and Taylor [53],Vizing [188])
INSTANCE: Undirected graph $G = (V, E)$, subsets $D_v \subseteq \mathbb{Z}^+$ (lists) for all $v \in V$, $D = \cup_{v \in V} D_v$, and positive integer K.
QUESTION: Does there exist an assignment of subsets $f : V \mapsto D$ such that,

(i). $f(v) \in D_v$,

(ii). $f(v) \neq f(w)$ for all $\{v, w\} \in E$, and

(iii). $|\cup_{v \in V} f(v)| \leq K$?

The problem Minimum Order List Coloring is $\mathcal{NP}$-complete, even for special graphs for which the graph coloring problem can be solved in linear time, e.g. for interval graphs [20].

For the general MO-FAP, an integer linear programming formulation has been presented by Aardal et al. [1]. For every vertex v and available frequency f a binary variable is introduced:

$$x_{vf} = \begin{cases} 1 & \text{if frequency } f \in D_v \text{ is assigned to vertex } v \\ 0 & \text{otherwise} \end{cases}$$

Moreover, a binary variable y_f denotes the use of frequency f:

$$y_f = \begin{cases} 1 & \text{if frequency } f \in D \text{ is used} \\ 0 & \text{otherwise} \end{cases}$$

Then, MO-FAP reads

$$\min \quad \sum_{f \in D} y_f \tag{2.1}$$

$$\text{s.t.} \quad \sum_{f \in D_v} x_{vf} = c_v \qquad \forall v \in V \tag{2.2}$$

$$x_{vf} + x_{wg} \leq 1 \qquad \forall \{v, w\} \in E, f \in D_v, g \in D_w : (|f - g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \tag{2.3}$$

$$x_{vf} \leq y_f \qquad \forall v \in V, f \in D_v \tag{2.4}$$

$$x_{vf} \in \{0, 1\} \qquad \forall v \in V, f \in D_v \tag{2.5}$$

$$y_f \in \{0, 1\} \qquad \forall f \in D \tag{2.6}$$

The constraints (2.2) model that c_v frequencies have to be assigned to connection $v \in V$. The forbidden combinations of frequencies are modeled by constraints (2.3), whereas (2.4) specifies that a y variable is set to one in case the corresponding frequency is used by the assignment. The objective (2.1) simply sums the use of the available frequencies.

2.3.2 Benchmark Instances

Benchmark instances for the MO-FAP are available via the CALMA project (see Section 2.2.4). For the minimum order instances, an overview of the results is presented

in Table 2.1. In all instances $c_v = 1$ for all $v \in V$. The average number of available frequencies is 40.

2.3.3 Lower Bounds and Exact Methods

In the framework of the CALMA project, Aardal et al. [1] applied integer programming techniques to the problem. They added cutting planes to the formulation (2.1)-(2.6). The used cutting planes are well known valid inequalities of the related vertex packing problem. Additional preprocessing techniques and specified branching strategies made it possible to solve large instances with up to 916 vertices to optimality. For all the tested instances, they proved the optimal values in this way. They compared the lower bounds obtained through the linear programming (LP) approach with bounds based on combinatorial arguments like cliques in the graph (cl.), the coloring number (col.) and the generalized coloring (gcol.). In all 4 cases tested, the LP provided the best bound.

Hurkens and Tiourine [91] computed lower bounds on the minimum order of any assignment. The lower bounds were derived through the detection of cliques in the constraint graph. By combination of the knowledge about several cliques the clique lower bound can be improved.

Finally, in Kolen, van Hoesel, and van der Wal [120] constraint satisfaction techniques were applied to the MO-FAP. For all but two of the instances the optimal solution is reported by combination of the lower and upper bounds generated by the technique.

2.3.4 Heuristics

Most heuristics for the MO-FAP were proposed in the framework of the CALMA project. Besides their lower bounding techniques, Tiourine, Hurkens, and Lenstra [91, 184] also applied several local search techniques, like simulated annealing (SA), tabu search (TS) and variable depth search (VDS) to the instances (see also Tiourine [182]). For 6 out of the 10 instances, they proved optimality combining the lower and upper bounds. A group from King's College London [25, 26] applied Tabu Search as well. In contrast with the tabu search approach of [184], the neighborhood function is less sophisticated, which explains the different performance of the algorithms. Moreover, in [26] a GEneral NETwork algorithm (GENET) for constraint satisfaction problems is applied to the same instances. They obtained optimal or near optimal solutions.

In the Master's Thesis of Warners [192], a potential reduction (PR) algorithm for the MO-FAP was introduced (see also Warners et al. [194, 193]). The algorithm is inspired by Karmarkar's interior point potential reduction approach to combinatorial optimiza-

instance	$\|V\|$	$\|E\|$	lower bounds						optimal	upper bounds									
			[1] cl.	[1] col.	[1] gcol.	[1] LP	[184] cl.	[120] CS	value (range)	[120] CS	[184] SA	[184] VDS	[184] TS	[26] TS	[26] GENET	[106] GA	[194] PR	[155] PR	[41] ES
CELAR 01	916	5,548	12	14	**16**	**16**	14	**16**	16	**16**	**16**	**16**	**16**	18	**16**	20	**16**	**16**	-
CELAR 02	200	1,235	**14**	**14**	**14**	**14**	**14**	**14**	14	**14**	**14**	**14**	**14**	**14**	**14**	**14**	**14**	**14**	**14**
CELAR 03	400	2,760	12	**14**	**14**	**14**	**14**	**14**	14	**14**	**14**	**14**	**14**	**14**	**14**	16	16	**14**	**14**
CELAR 04	680	3,967	-	-	-	**46**	**46**	**46**	46	**46**	**46**	**46**	**46**	**46**	**46**	**46**	**46**	**46**	-
CELAR 11	680	4,103	20	20	20	**22**	**22**	**22**	22	**22**	24	24	**22**	24	24	32	-	**22**	-
GRAPH 01	200	1,134	-	-	-	**18**	**18**	**18**	18	**18**	-	-	**18**	**18**	**18**	20	**18**	**18**	**18**
GRAPH 02	400	2,245	-	-	-	**14**	**14**	**14**	14	**14**	-	-	**14**	16	**14**	16	**14**	**14**	**14**
GRAPH 08	680	3,757	-	-	-	-	16	-	16-18	-	-	-	20	24	22	-	18	18	-
GRAPH 09	916	5,246	-	-	-	-	**18**	**18**	18	**18**	-	-	22	22	22	28	**18**	**18**	-
GRAPH 14	916	4,638	-	-	-	-	**8**	-	8	10	-	-	10	12	-	14	10	**8**	-

Table 2.1: Minimum Order benchmark instances CALMA project. Framed values indicate the optimal value. Some of the results are only mentioned in [183].

tion problems (cf. [105, 108, 109]). Pasechnick [155] improved the performance of the algorithm, and proved the optimal value of the instance `GRAPH 14`.

Kapsalis, Rayward-Smith and Smith [107] (see also [106]) applied a genetic algorithm to the instances. The results are less satisfactory than of the other algorithms. Only for 2 instances they obtained the optimal solution. Crisan and Mühlenbein [41] applied evolutionary search (ES) to MO-FAP. Evolutionary search consists of the repeatedly mutation of a solution according to a certain mutation operator. They investigated the performance of an evolutionary search algorithm, and analyzed the search space in order to obtain some information on the difficulty of the instances. Computational results were carried out on the CALMA instances. They concluded from the analysis that there is far less relationship between two good frequency assignment plans, than there is for instance between two good tours in the traveling salesman problem. This implies that local search techniques will have more difficulties to reach the optimum for MO-FAP, than comparable heuristics for the traveling salesman problem. Their computational results with evolutionary search are comparable with the results of the tabu search, simulated annealing or variable depth search in [184].

Finally, Cuppini [42] applied a genetic algorithm to the minimum order problem. In contrast with other genetic algorithms for FAPs, an assignment is represented by $|D|$ genes of $N = \sum_{v \in V} c_v$ elements (in most genetic algorithms $|V|$ genes of size $|D_v|$ are used to represent an assignment). Computational results are only reported for a small example.

2.3.5 Conclusions

Summarized, good heuristics for the MO-FAP seem to be most well known local search techniques, as well as the potential reduction technique based on interior point methods. For the local search algorithms the quality of the solution heavily depends on the neighborhood that is used. To guarantee optimality, lower bounding techniques based on integer and quadratic programming can be applied successfully.

2.4 Minimum Span Frequency Assignment

2.4.1 Problem Definition

In the minimum span frequency assignment problem (MS-FAP), the problem is to assign frequencies in such a way that no unacceptable interference occurs, and the difference between the maximum and minimum used frequency, the span, is minimized. Formally

we can describe the problem as follows:

Minimum Span Frequency Assignment (MS-FAP)
INSTANCE: Undirected graph $G = (V, E)$, $\{v, v\} \in E$, for all $v \in V$, sets $T_{vw} \subset \mathbb{Z}$, $\{v, w\} \in E$, $0 \in T_{vw}$, demand $c_v \in \mathbb{Z}^+$, domain subsets $D_v \subseteq \mathbb{Z}^+$ for all $v \in V$, $D = \cup_{v \in V} D_v$, and positive integer K.
QUESTION: Does there exist an assignment of subsets $f : V \mapsto 2^D$ such that,

(i). $|f(v)| = c_v$,

(ii). $f(v) \subseteq D_v$,

(iii). $|\bar{f} - \bar{g}| \notin T_{vw}$ for all $\{v, w\} \in E$, $\bar{f} \in f(v)$, $\bar{g} \in f(w)$, $v \neq w$ or $\bar{f} \neq \bar{g}$, and

(iv). $\max \cup_{v \in V} f(v) - \min \cup_{v \in V} f(v) \leq K$?

In case $D_v = \mathbb{Z}^+$ and $T_{vw} = \{0\}$, the problems MO-FAP and MS-FAP are equivalent [78]. In general, however, there exist examples in which neither a minimum order assignment with minimum span, nor a minimum span assignment with minimum order exists. The special case of minimum span T-coloring attracted a lot of attention in the literature, due to its relation with both the coloring problem and the MS-FAP. Roberts [160] presented a survey on T-coloring problems. More recently, theoretical results on T-coloring in the context of the MS-FAP are obtained by Griggs and Liu [76] and Liu [133]. A survey on frequency assignment problems with emphasis to the relation with graph theory by Murphey, Pardalos and Resende will appear in [147].

Several other authors have investigated coloring problems related to MS-FAP. For instance, Kubale [125] presented lower and upper bounds, and considered special cases for a graph-coloring problem related to the MS-FAP in which we have to color each vertex v with c_v consecutive colors (interval coloring). Like in the MS-FAP the span of the assignment must be minimized. In Kubale [126] complexity results are presented for another minimum span coloring problem with forbidden colors (minimum span list coloring).

The interval T-coloring problem has been studied by De Werra and Gay [195]. In the interval T-coloring problem we have to assign c_v consecutive colors to v in such a way that the assignment does not violate the sets T_{vw}. This problem is equivalent to an asymmetric MS-FAP, i.e. a FAP in which, instead of $|f - g| \notin T_{vw}$, we have to satisfy $f - g \notin T_{vw}$, where $T_{vw} \subset \mathbb{Z}$ may also contain negative numbers, and is not necessarily symmetric with respect to 0. De Werra and Gay [195] derived upper bounds on the minimum span of the asymmetric MS-FAP. Moreover, they apply a heuristic based on the graph coloring algorithms of Brélaz [28] on randomly generated instances (with in some cases Euclidean distances).

For the general case, an integer programming formulation similar to (2.1)-(2.6) reads

$$\min \quad z_{\max} - z_{\min} \tag{2.7}$$

$$\text{s.t.} \quad \sum_{f \in D_v} x_{vf} = c_v \qquad \forall v \in V \tag{2.8}$$

$$x_{vf} + x_{wg} \leq 1 \qquad \forall \{v, w\} \in E, f \in D_v, g \in D_w : (|f - g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \tag{2.9}$$

$$x_{vf} \leq y_f \qquad \forall v \in V, f \in D_v \tag{2.10}$$

$$z_{\max} \geq f y_f \qquad \forall f \in D \tag{2.11}$$

$$z_{\min} \leq f_{\max} - (f_{\max} - f) y_f \qquad \forall f \in D \tag{2.12}$$

$$x_{vf} \in \{0, 1\} \qquad \forall v \in V, f \in D_v \tag{2.13}$$

$$y_f \in \{0, 1\} \qquad \forall f \in D \tag{2.14}$$

$$z_{\min}, z_{\max} \in \mathbb{Z}^+ \tag{2.15}$$

where $f_{\max} = \max_{f \in D} f$ the maximum available frequency, and $z_{\min}$ and $z_{\max}$ are two additional variables for the minimum and maximum used frequency, respectively. The constraints (2.11) and (2.12) guarantee that these variables are set to the right values.

Several other ways to model the objective can be applied. For instance, instead of the y and z variables we can use binary variables l_f and u_f:

$$l_f = \begin{cases} 1 & \text{if } f \in D \text{ is the smallest frequency that is used} \\ 0 & \text{otherwise} \end{cases}$$

and

$$u_f = \begin{cases} 1 & \text{if } f \in D \text{ is the largest frequency that is used} \\ 0 & \text{otherwise} \end{cases}$$

Then the objective (2.7) can be replaced by

$$\min \quad \sum_{f \in D} f(u_f - l_f)$$

and the constraints (2.10)-(2.12) have to be replaced by

$$\sum_{f \in D} l_f = 1$$

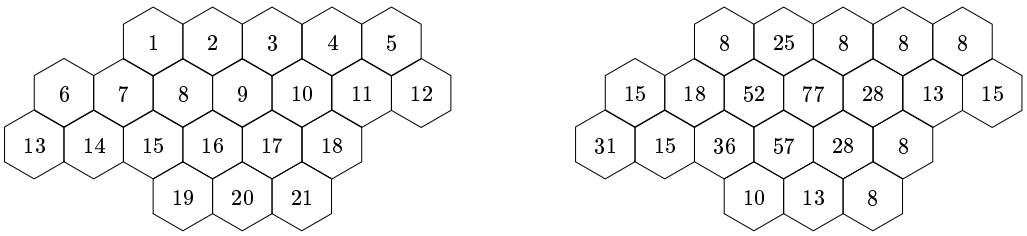
$$\sum_{f \in D} u_f = 1$$

$$x_{vf} + l_g \leq 1 \qquad \forall v \in V, f, g \in D_v, f < g$$

$$x_{vf} + u_g \leq 1 \qquad \forall v \in V, f, g \in D_v, f > g$$

In case $D_v = D$ for all $v \in V$ and D consists of consecutive numbers, minimization of the span is equivalent with minimization of the maximum frequency used. This implies that in that case the $z_{\min}$ variable (or the l_f variables) can be left out of the formulation.

2.4.2 Benchmark Instances



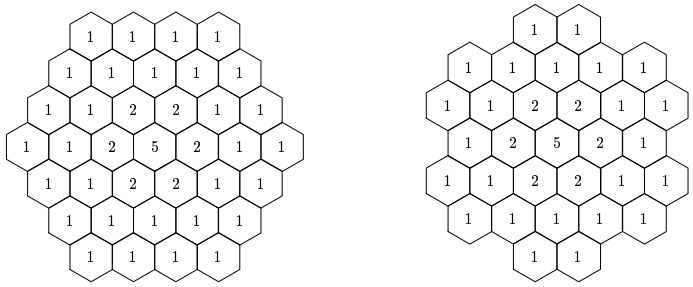
(a) network structure (b) instance P1

Figure 2.1: Philadelphia instances.

To test proposed algorithms a number of benchmark instances are available. In 1973, Anderson [12] introduced the Philadelphia instances. The original instance and certain variants are widely used afterwards to substantiate algorithms and lower bounds for the MS-FAP. The Philadelphia instances are characterized by 21 hexagons denoting the cells of a cellular phone network around Philadelphia (see Figure 2.1(a)). Until recently, it was common practice to model wireless phone networks as hexagonal cell systems. For each cell, a demand c_v is given. Figure 2.1(b) shows the demand for the original instance `P1`, whereas Table 2.2 contains the demand vectors of all instances. In conformity with [186], the instances are denoted by `P1-P9`. Some of them are also appointed in [92] as `E3-E9`. In the basic model, interference of cells is characterized by a co-channel reuse distance d. No interference occurs if and only if the centers of two cells have mutual distance $\geq d$. In case the mutual distance is less than d (normalized by the radius of the cells), it is not allowed to assign the same frequency to both cells. This pure co-channel case is generalized by replacing the reuse distance d by a series of non-increasing values $d^0, \ldots, d^k$ and corresponding forbidden sets $T^0 \subseteq \ldots \subseteq T^k$. The following relation holds:

$$T_{vw} = T^{j-1} \text{ whenever } d^j \leq d_{vw} < d^{j-1}, j \in \{1, \ldots, k\}$$

where d_{vw} is the distance between the cell centers. For the Philadelphia instances, the sets T^j are taken as $T^j = \{0, \ldots, j\}$. For instance, `P1` the values $d^0, \ldots, d^5$ are $2\sqrt{3}, \sqrt{3}, 1, 1, 1, 0$. So, frequencies assigned to the same site should be separated by at least 4 other frequencies, whereas frequencies assigned to adjacent sites should be at a distance of at least 2, and frequencies assigned to a second and third 'ring' of cells should still differ (see Figure 2.2). For the other instances, the reuse distances are given in Table 2.2. The domains D_v are simply $\mathbb{Z}^+$, in which case minimization of the span is equivalent to minimization of the maximum used frequency. Note that, there is a difference of one between minimum span and maximum used frequency. Table 2.3 shows the results obtained on the Philadelphia instances.



(a) P1, P3, P5, P7, P9 (b) P2, P4, P6

Figure 2.2: Reuse distances Philadelphia instances. The values denote the minimum separation distance in relation to the central cell.

A second set of benchmark instances, that are not defined on a hexagonal grid, are available via the CALMA project [32]. In Table 2.4 the characteristics and the results are presented.

2.4.3 Lower Bounds and Exact Methods

A lot of research has been devoted to lower bounds on the span for the Philadelphia instances. The first non-trivial lower bounds for the MS-FAP, with $T_{vw} = \{0, \ldots, k\}$ for some $k \in \mathbb{Z}^+$, and $D_v = \mathbb{Z}^+$, were presented by Gamst [60]. He applied several lower bounds based on graph theory (GT) to instance `P1`. The most important bound of Gamst is the clique bound. Let the vertices $S \subseteq V$ form a complete subgraph (clique) in the graph G. Let $T_{\min} = \min_{\{v,w\} \in E[S]} |T_{vw}|$ (with the assumption that T_{vw} is a set of consecutive values). Then the span of any assignment, $sp(G) \geq T_{\min}(\sum_{v \in S} c_v - 1)$. Two more sophisticated bounds are given in [60]. Although, Gamst only computed the lower bounds for `P1`, Table 2.3 shows the best lower bound for all instances.

instance	demand vector c_v	reuse distances
P1 (E3)	$(8,25,8,8,8,15,18,52,77,28,13,15,31,15,36,57,28,8,10,13,8)$	$(2\sqrt{3},\sqrt{3},1,1,1,0)$
P2 (E4)	$(8,25,8,8,8,15,18,52,77,28,13,15,31,15,36,57,28,8,10,13,8)$	$(\sqrt{7},\sqrt{3},1,1,1,0)$
P3 (E5)	$(5,5,5,8,12,25,30,25,30,40,40,45,20,30,25,15,15,30,20,20,25)$	$(2\sqrt{3},\sqrt{3},1,1,1,0)$
P4 (E6)	$(5,5,5,8,12,25,30,25,30,40,40,45,20,30,25,15,15,30,20,20,25)$	$(\sqrt{7},\sqrt{3},1,1,1,0)$
P5 (E7)	$(20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20)$	$(2\sqrt{3},\sqrt{3},1,1,1,0)$
P6	$(20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20,20)$	$(\sqrt{7},\sqrt{3},1,1,1,0)$
P7 (E8)	$(16,50,16,16,16,30,36,104,154,56,26,30,62,30,72,114,56,16,20,26,16)$	$(2\sqrt{3},\sqrt{3},1,1,1,0)$
P8	$(8,25,8,8,8,15,18,52,77,28,13,15,31,15,36,57,28,8,10,13,8)$	$(2\sqrt{3},2,1,1,1,0)$
P9 (E9)	$(32,100,32,32,32,60,72,208,308,112,52,60,124,60,144,228,112,32,40,52,32)$	$(2\sqrt{3},\sqrt{3},1,1,1,0)$

Table 2.2: Characteristics Philadelphia benchmark instances.

instance	lower bounds				optimal	upper bounds								
	[60]	[95, 96]	[92, 173]	[179]	value	[92]	[92]	[92]	[186]	[173]	[171]	[179]	[179]	[191]
	GT	TSP	div.	GT	(range)	Best Seq.	TS	SA	GA		Seq.	GSP1	GSP2	LS
P1	413	426	426	426	426	447	428	428	426	426	459	440	450	432
P2	413	426	426	426	426	475	429	438	426	-	446	436	444	-
P3	245	-	257	252	257	284	269	260	258	257	282	291	273	263
P4	245	252	252	252	252	268	257	259	253	252	268	273	268	-
P5	239	-	239	177	239	250	240	239	239	-	-	-	-	-
P6	139	177	178	177	178-188	230	188	200	198	-	-	-	-	-
P7	830	-	855	855	855-856	894	858	858	856	-	-	-	-	-
P8	449	-	524	427	524-527	592	535	546	527	-	-	-	-	-
P9	1,664	-	1,713	1,713	1,713-1,724	1,800	1,724	1,724	-	-	-	-	-	-

Table 2.3: Results Philadelphia benchmark instances. Framed values indicate the optimal value.

instance	characteristics		lower bounds			upper bounds				
	$\|V\|$	$\|E\|$	[1] LP	[184] QP	[120] CS	[184] TS	[25] TS	[26] GENET	[194] PR	[155] PR
CELAR 05	400	2,598	792	792	792	792*	792	792	792	792
GRAPH 03	200	1,134	-	-	380	380	-	-	-	380
GRAPH 04	400	2,244	-	-	394	394	-	-	-	394
GRAPH 10	680	3,907	-	-	394	394	-	-	394	394

Table 2.4: Minimum Span benchmark instances CALMA project. *Instead of tabu search, simulated annealing is applied to this instance.

Only few researchers have succeeded in finding better bounds. The best were obtained by Janssen and Kilakos [95, 96] in their study of the minimum span problem from a polyhedral point of view. The traveling salesman problem (TSP) on a related graph G' can be seen as a relaxation of the MS-FAP, which means that every lower bound for the TSP, is a lower bound for the corresponding MS-FAP as well. The relation between MS-FAP and TSP was first observed by Raychaudhuri [158], and used by Roberts [160] and Smith and Hurley [172]. Let G' be a weighted complete graph with same vertex set as G. Let the weights $w_{v_iv_j} = 0$ if $\{v_i, v_j\} \notin E$, and $w_{v_iv_j} = |T_{v_iv_j}| + 1$ if $\{v_i, v_j\} \in E$. Let $H(G')$ denote the length of the shortest Hamiltonian path in G'. Then $sp(G) \geq H(G')$. Since, the minimum spanning tree $S(G')$ is a lower bound for the shortest Hamiltonian path, it also holds that $sp(G) \geq S(G')$. Janssen and Kilakos relaxed the TSP formulation to the edge cover polytope and study the polyhedral structure of the dual of this problem. In addition, they also study the polyhedral structure of the dual of the TSP linear programming relaxation. For the most studied Philadelphia problem, P1, they prove a lower bound of 426. Combined with an upper bound of 426 this implies that this problem is solved [95]. In [97], they also studied the polyhedral structure of the dual of the tile covers formulation for the MS-FAP.

Lower bounds based on subgraphs and preprocessing ideas are presented by Smith and Hurley [172] (see also [173]). Every lower bound / optimal span on a subgraph of G provides a lower bound on the span of G itself. Preprocessing ideas incorporate the deletion of vertices with (almost) the same neighborhood, and the deletion of vertices for which they can prove that there is always an assignment possible within the lower bound spectrum. Recently, Allen, Smith, and Hurley [11] derived new lower bounds by integer programming techniques. They extended the integer programming formulation for the Hamiltonian path problem, with additional constraints and variables that represent the MI-FAP. Application of integer programming techniques, like branch-and-bound and Lagrangean relaxation resulted in improved lower bounds for a small instance.

Other lower bounds are derived by Sung and Wong [179] and Tcha, Chung, and Choi [180].

In [179] a new lower bound for the MS-FAP is presented, based on similar arguments as the bounds of Gamst. For most instances, the lower bound is as strong as the TSP bound. In [180] one of the lower bounds of Gamst [60] is extended. On a variant of instance `P1` they prove that the new lower bound indeed can improve the lower bound of Gamst.

Finally, for the CALMA benchmark instances (cf. Table 2.4), researchers applied the same methods as for the MO-FAP. Hurkens and Tiourine [91] applied the clique lower bound techniques, whereas Aardal et al. [1] applied branch-and-cut based on the formulation (2.7)-(2.15) to instance `CELAR 05`. It turned out that the CALMA instances are quite simple to solve to optimality. In fact, in Kolen, van Hoesel, and van der Wal [120], optimality results are reported for all instances via constraint satisfaction (CS) techniques.

The same formulation (2.7)-(2.15) is the topic of a study by Giortzis and Turner [65]. They applied branch-and-bound with a branching priority rule to an instances with 58 vertices ($c_v = 4$) and 29 available frequencies. They proved that the optimal solution needs 16 frequencies.

2.4.4 Heuristics

The first heuristics for the MS-FAP (e.g., Philadelphia instances) are proposed as early as the 1970s. In Box [27] and Zoellner and Beall [197] the first constructive heuristics are introduced. The frequencies are assigned to the vertices according to some order of the vertices. In Sivarajan, McEliece and Ketchum [171] several variants of the algorithm are tested on 13 Philadelphia instances. Quite a lot of the variants turned out to be trivial. For the remaining instances, `P1-P4` the results are reported in Table 2.3. None of the 8 tested variants outperformed the other ones.

In Smith, Hurley and Thiel [173], the derived lower bounds are combined with a heuristic. The heuristic first assigns a subgraph in the graph, and afterwards tries to extend the assignment to a complete assignment with the same span. If such an assignment is not possible they extend the subgraph with an additional vertex, and repeat the procedure. Optimal solutions are presented for three Philadelphia instances. In [92], the same authors describe the software system FASoft, a planning tool for frequency assignment based on these results. In this paper they describe several sequential assignment algorithms (like those by Sivarajan, McEliece and Ketchum [171], as well as improvement heuristics like tabu search (TS), simulated annealing (SA) and genetic algorithms (GA). For the 48 variants of sequential assignment algorithms the best one is reported in Table 2.3. For the genetic algorithms no results are reported in [92]. Valenzuela, Hurley and Smith [186] applied a GA to these instances. Each assignment is represented by a permutation of the vertices. An assignment is obtained by assigning frequencies to the vertices in a greedy way according to the permutation. They tested the algorithm on the Philadelphia

instances P1-P8. In three cases the optimal solution was found.

Besides their lower bound, Sung and Wong [179] also described a heuristic that provides an optimal solution in a special case. They prove that their sequential packing algorithm provides an optimal span in case only co-channel constraints are taking into account, and the hexagonal cell network contains at most 3 stripes, i.e., it can be represented by 3 rows of hexagonal cells. For cases with adjacent channel constraints, the algorithm is generalized. Two versions of the algorithm, GSP1 and GSP2, are tested on the instances P1-P4 (see Table 2.3) and some easier variants.

Wang and Rushforth [191] discuss a local search method for the MS-FAP. First the vertices are assigned frequencies according to some sequence. Next, they exchange the assignment of two vertices as long as the objective improves. In case no improvement is possible anymore, also non-deterioration is allowed to escape from local minima. They tested their algorithm on two Philadelphia instances (see Table 2.3) and on instances presented in Kim and Kim [114]. In [114] a two phase heuristic is introduced to solve the minimum span problem. They assume a hexagonal grid and use patterns consisting of a number of cells to which we can assign the same frequency. The algorithm is tested on randomly generated instances.

In the context of the minimum span problem, we also should mention the work of Lanfear [130]. In his comprehensive overview of frequency assignment, four algorithms for the MS-FAP are proposed: an exact search algorithm (branch-and-bound), a simulated annealing algorithm, a tabu search algorithm and an algorithm based on sequencing the vertices are described. The simulated annealing algorithm can only be applied to instances with constraints restricted to co-channel and adjacent-channel interference (i.e., $d_{vw} \in \{1, 2\}$ for all $\{v, w\} \in E$).

For the CALMA instances, all heuristics performed equally, and found the optimal solution. Tabu search was applied by Tiourine, Hurkens, and Lenstra [91, 184]. In [26], TS and GENET results are reported for CELAR 05. The potential reduction (PR) method by Warners [192] was used to solve both CELAR 05 and GRAPH 10. Pasechnik [155] also applied potential reduction to the minimum span problems. Next to it, he solved the minimum order problems as minimum span instances. For GRAPH 01 he could prove that the minimal span equals 408, whereas for the other instances lower and upper bounds were derived.

2.4.5 Conclusions

Summarized, for the MS-FAP without specific domains, good lower bounds are provided by several authors. The lower bounds are tested extensively on the Philadelphia instances. The heuristic techniques proposed for these problems seem to be less accurate in providing

optimal solutions in all cases. In cases where specific domains are given, the benchmark instances are less challenging. In all cases the best solution is found with the applied heuristics, whereas also lower bounds are available to guarantee optimality. More difficult benchmark instances are necessary to distinguish among the heuristics.

2.5 Minimum Blocking Frequency Assignment

2.5.1 Problem Definition

In case all assignments contain some unacceptable interference, we can decide to find a partial assignment that minimizes the overall blocking probability. In the minimum blocking frequency assignment problem (MB-FAP), the problem is to assign frequencies in such a way that no unacceptable interference occurs and the overall blocking probability of the network is minimized. More formally the problem is defined

Minimum Blocking Frequency Assignment (MB-FAP)
INSTANCE: Undirected graph $G = (V, E)$, $\{v, v\} \in E$, for all $v \in V$, sets $T_{vw} \subset \mathbb{Z}$, $\{v, w\} \in E$, $0 \in T_{vw}$, demand $c_v \in \mathbb{Z}^+$, domain subsets $D_v \subseteq \mathbb{Z}^+$ for all $v \in V$, $D = \cup_{v \in V} D_v$, non-increasing blocking function $b_v : \mathbb{Z}_0^+ \mapsto \mathbb{Z}_0^+$ for all $v \in V$, and positive integer K.
QUESTION: Does there exist an assignment of subsets $f : V \mapsto 2^D$ such that,

(i). $|f(v)| \leq c_v$,

(ii). $f(v) \subseteq D_v$,

(iii). $|\bar{f} - \bar{g}| \notin T_{vw}$ for all $\{v, w\} \in E$, $\bar{f} \in f(v)$, $\bar{g} \in f(w)$, $v \neq w$ or $\bar{f} \neq \bar{g}$, and

(iv). $\sum_{v \in V} b(|f(v)|) \leq K$?

The special case in which $c_v = 1$, $b_v(0) = 1$, $b_v(1) = 0$, $|D_v| = 1$, $D_v = D_w$ for all $v, w \in V$, and $T_{vw} = \{0\}$ for all $\{v, w\} \in E$, is equivalent with the maximum independent set problem. As a consequence, MB-FAP is $\mathcal{NP}$-complete in general. An integer programming formulation (with nonlinear objective) for this problem reads

$$\min \quad \sum_{v \in V} b_v(m_v) \tag{2.16}$$

$$\text{s.t.} \quad m_v = \sum_{f \in D_v} x_{vf} \leq c_v \qquad \forall v \in V \tag{2.17}$$

$$x_{vf} + x_{wg} \leq 1 \qquad \forall \{v, w\} \in E, f \in D_v, g \in D_w :$$

$$(|f-g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \qquad (2.18)$$

$$x_{vf} \in \{0,1\} \qquad \forall v \in V, f \in D_v \qquad (2.19)$$

The constraints (2.17) model that at most c_v frequencies should be assigned to $v \in V$. The value m_v is only used to simplify the objective (2.16) that minimizes the overall blocking probability. The objective (2.16) is a generalized version of the objective of Chang and Kim [36]. They model the MB-FAP as a non-linear combinatorial optimization problem. Their objective function really represents the blocking probability. In conformity with Chang and Kim, let λ_v denote the traffic demand in Erlang for cell v, and m_v the number of assigned channels. Then the cell blocking probability of cell v is given by the Erlang B formula as

$$B(\lambda_v, m_v) = \left(\sum_{k=0}^{m_v} \frac{(\lambda_v)^k}{k!} \right)^{-1} \frac{(\lambda_v)^{m_v}}{m_v!}$$

The weighted average blocking probability for a vertex v is then given by

$$b_v(m_v) = w_v B(\lambda_v, m_v)$$

with $w_v = \lambda_v / \sum_{v \in V} \lambda_v$ the traffic weighting factor. Since, the function $B(\lambda_v, m_v)$ is strictly decreasing and convex in m_v, we can linearize the objective function by the introduction of coefficients $\alpha_{vm} := B(\lambda_v, m) - B(\lambda_v, m-1) < 0$, and the binary variables y_{vm} denoting

$$y_{vm} = \begin{cases} 1 & \text{if at least } m \leq c_v \text{ frequencies are assigned to } v \in V \\ 0 & \text{otherwise} \end{cases}$$

Then, the objective (2.16) reads

$$\min \quad \sum_{v \in V} w_v \left(1 + \sum_{m=1}^{c_v} \alpha_{vm} y_{vm} \right) \qquad (2.20)$$

and the constraint (2.17) reads

$$\sum_{m=1}^{c_v} y_{vm} = \sum_{f \in D_v} x_{vf} \leq c_v \qquad \forall v \in V \qquad (2.21)$$

Note that, $y_{vm} = 1$ implies $y_{vm-1} = 1$, since the function $B(\lambda_v, m_v)$ is strictly convex, which implies that α_{vm} strictly increases over m.

The same objective is used by Mathar and Mattfeldt [141]. In all remaining articles the objective is simplified to $b_v(m) = c_v - m$, i.e., the unsatisfied demand is minimized, or equivalently the number of assigned frequencies is maximized. Therefore, this problem is also called the maximum service frequency assignment problem.

2.5.2 Lower Bounds and Exact Methods

Chang and Kim [36] first linearize (2.16) to (2.20). Next, they generate a number of patterns (i.e., a pair (S, f), subset $S \subset V$ and a frequency $f \in D$ that can be assigned without interference to all the vertices $v \in S$ simultaneously). Then, the problem can be remodeled in terms of these patterns, and Lagrangean Relaxation is applied to the new formulation. Furthermore, they describe a grade-of-service (GoS) updating heuristic. They tested their algorithm on randomly generated instances based on a 7×7 hexagonal grid network.

Besides the co-channel and adjacent-channel constraints, represented by (2.18), Fischetti et al. [54] also take into account that the overall interference affecting a cell has to be limited to a value L,

$$\sum_{u \in V} \sum_{g \in D_u} p_{vufg} x_{ug} \leq L + M(1 - x_{vf}) \qquad \forall v \in V, f \in D_v \qquad (2.22)$$

where p_{vufg} is the interference level of the combination (v, f) and (u, g), and M is a big constant with respect to the interference levels. In case frequency $f \in D_v$ is selected the total level of interference should be below L, in case $f \in D_v$ is not selected the constraint is redundant. The value L corresponds with the signal-to-noise ratio and is for example set to 0.125687 (9 dB). In [54] only co-channel and adjacent-channel interference is taken into account. Let $I_{vu} \geq 0$ denote the real interference level by use of the same frequency for v and u, and let NFD denote the *Net Filter Discriminator*, a reduction factor for adjacent frequencies. Then (2.22) reduces to

$$\sum_{u \in V} \left(I_{vu} x_{uf} + \frac{I_{vu}}{NFD} (x_{uf-1} + x_{uf+1}) \right) \leq L + M(1 - x_{vf}) \;\; \forall v \in V, f \in D_v \qquad (2.23)$$

In [54] the problem is solved with Branch and Cut. Their instances are obtained from CSELT (a research laboratory connected to TIM, one of the Italian mobile radio system managers) and contain up to 203 vertices. Not all instances can be solved to optimality. The same instances have been studied by Mannino and Sassano [140]. They present an enumeration scheme, within the context of a core search. They assign first the (difficult) core of the problem, and afterwards extend the assignment to the complete problem, without additional interference. Their algorithm outperforms the Branch and Cut approach

of Fischetti et al. on all instances, both in time and optimality results. In [140], the adaptive core search algorithm is also tested on several instances from an Italian broadcasting company. The overall interference (2.22) is not taken into account in these instances.

The problem of minimizing the unsatisfied demand is also studied by Jaumard, Marcotte, Meyer and Vovor [101] (see also [99, 102]). Besides the demand c_v, they also take into account a minimum number of required frequencies $\underline{c}_v$, resulting in the additional constraint

$$\sum_{f \in D_v} x_{vf} \geq \underline{c}_v \qquad \forall v \in V \tag{2.24}$$

They compare 3 different integer programming formulations, one equivalent to (2.16)-(2.19) and the formulation of Mehrotra and Trick [143] for the graph coloring problem, and two set-covering formulations. They compare the formulations with respect to the quality of the linear programming relaxation. For the best formulation, one of the set-covering formulations, the integrality gap remains significant. They use column generation techniques to solve the linear programming relaxation, and present an efficient branching scheme to be used within a branch and cut framework. They report results on two medium-sized problems of Bell Mobility.

In Giortzis and Turner [65], 5 instances with 4 to 58 vertices and in between 5 and 29 available frequencies are solved with standard branch-and-bound. To improve the performance of the algorithm, they applied a special branching priority on the variables x_{vf}.

Finally, the MB-FAP is also the topic of the papers by Kazantzakis, Demestichas and Anagnostou [112] and Rouskas, Kazantzakis and Anagnostou [163]. They present an integer linear programming formulation similar to (2.16)-(2.19) for the problem. They solve the linear programming relaxation and add inequalities that the objective has to be integral. However, in case the objective value is integral, the solution can still be fractional. The search for an integral solution is done via an exhaustive search of the solution space of an integer quadratic program representing all integral solutions with the given objective value. Computational results are reported on a small test problem.

2.5.3 Heuristics

Only one heuristic approach is known for the MB-FAP. Mathar and Mattfeldt [141] applied simulated annealing to the MB-FAP with the same objective as Chang and Kim [36]. They only took into account the co-channel interference. The quality of their solutions is examined through the use of special network structures for which optimal solutions can be computed efficiently.

2.5.4 Conclusions

Concluding, most approaches to solve the MB-FAP deal with exact solution techniques. This direction is inspired by the relation with the maximum independent set problem which belongs to the standard problems in combinatorial optimization, and therefore, has been the topic of many studies. Although benchmark instances are not available, the results show that reasonable large real-life instances can be solved to optimality with integer programming techniques and search algorithms in which combinatorial arguments are incorporated.

2.6 Minimum Interference Frequency Assignment

2.6.1 Problem Definition

Besides the approaches in which the maximum interference level is minimized, another approach is given by the minimization of the total sum of interference levels. In the minimum interference frequency assignment problem (MI-FAP), we have to assign frequencies from a limited number of available frequencies in such a way that the total sum of weighted interference is minimized. Formally, the problem can be defined as

Minimum Interference Frequency Assignment
INSTANCE: Undirected graph $G = (V, E)$, $\{v, v\} \in E$, for all $v \in V$, sets $T_{vw} \subset \mathbb{Z}$, $\{v, w\} \in E$, $0 \in T_{vw}$, demand $c_v \in \mathbb{Z}^+$, domain subsets $D_v \subseteq \mathbb{Z}^+$ for all $v \in V$, $D = \cup_{v \in V} D_v$, penalty values $p_{vwfg} \in \mathbb{Z}^+$, for all $\{v, w\} \in E$, $f \in D_v$, $g \in D_w$, and positive integer K.
QUESTION: Does there exist an assignment of subsets $f : V \mapsto 2^D$ such that,

(i). $|f(v)| = c_v$,

(ii). $f(v) \subseteq D_v$, and

(iii). $$\sum_{\{v,w\} \in E} \sum_{\substack{\bar{f} \in f(v), \bar{g} \in g(w) \\ (v \neq w) \vee (\bar{f} \neq \bar{g})}} p_{vw\bar{f}\bar{g}} \delta(|\bar{f} - \bar{g}| \in T_{vw}) \leq K \ ?$$

Here, $\delta(A)$ is the Kronecker delta function which is equal to one in case the logical condition A is true and zero otherwise.

In many cases the MI-FAP is used as a subroutine to find the minimum span of a FAP. In this special case we would like to find an interference-free assignment to the vertices,

i.e., $K = 0$. This problem is also known as the Feasibility Frequency Assignment problem.

Feasibility Frequency Assignment
INSTANCE: Undirected graph $G = (V, E)$, $\{v, v\} \in E$, for all $v \in V$, sets $T_{vw} \subset \mathbb{Z}$, $\{v, w\} \in E$, $0 \in T_{vw}$, demand $c_v \in \mathbb{Z}^+$, and domain subsets $D_v \subseteq \mathbb{Z}^+$ for all $v \in V$, $D = \cup_{v \in V} D_v$.
QUESTION: Does there exist an assignment of subsets $f : V \mapsto 2^D$ such that,

(i). $|f(v)| = c_v$,

(ii). $f(v) \subseteq D_v$, and

(iii). $|\bar{f} - \bar{g}| \notin T_{vw}$ for all $\{v, w\} \in E$, $\bar{f} \in f(v)$, $\bar{g} \in f(w)$, $v \neq w$ or $\bar{f} \neq \bar{g}$?

An integer programming formulation for the MI-FAP can be given by the introduction of new binary variables z_{vfwg}, for all $\{v, w\} \in E$, $f \in D_v$, $g \in D_w$, with $|f - g| \in T_{vw}$, and either $v \neq w$ or $f \neq g$:

$$z_{vwfg} = \begin{cases} 1 & \text{if both } x_{vf} = 1 \text{ and } x_{wg} = 1 \\ 0 & \text{otherwise} \end{cases}$$

Then MI-FAP reads

$$\min \sum_{\{v,w\} \in E} \sum_{\substack{f \in D_v, g \in D_w \\ |f-g| \in T_{vw} \wedge (v \neq w \vee f \neq g)}} p_{vwfg} z_{vwfg} \tag{2.25}$$

$$\text{s.t.} \quad \sum_{f \in D_v} x_{vf} = c_v \qquad \forall v \in V \tag{2.26}$$

$$x_{vf} + x_{wg} \leq 1 + z_{vwfg} \qquad \forall \{v, w\} \in E, f \in D_v, g \in D_w : (|f - g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \tag{2.27}$$

$$x_{vf} \in \{0, 1\} \qquad \forall v \in V, f \in D_v \tag{2.28}$$

$$z_{vwfg} \in \{0, 1\} \qquad \forall \{v, w\} \in E, f \in D_v, g \in D_w : (|f - g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \tag{2.29}$$

Constraints (2.27) model the fact that both f and g can be assigned to v and w if and only if z_{vwfg} is equal to one, which implies an additional penalty in the objective (2.25). Since we assume $p_{vwfg} > 0$, the z variables equal 0 in case only one of the x variables

in (2.27) is set to 1. In case $p_{vwfg} < 0$, the constraints

$$z_{vwfg} \leq x_{vf} \qquad \forall \{v,w\} \in E, f \in D_v, g \in D_w : \tag{2.30}$$

$$(|f-g| \in T_{vw}) \wedge ((f \neq g) \vee (v \neq w)) \tag{2.31}$$

have to be added to the formulation.

Another way to model (2.27) is by the introduction of the variables z_{vwfg} for all $\{v,w\} \in E$, $f \in D_v$, $g \in D_w$ and the constraints

$$\sum_{g \in D_w} z_{vwfg} = c_w x_{vf} \qquad \forall \{v,w\} \in E, f \in D_v \tag{2.32}$$

In case $x_{vf} = 0$, then the constraints (2.32) enforce that all the variables z_{vwfg} are set to 0 as well. In case $x_{vf} = 1$, the constraints (2.32) guarantee that exactly c_w variables z_{vwfg} are set to 1; the variables z_{vwfg} with $x_{wg} = 1$. The model with the constraints (2.32) (and $c_v = 1$ is the topic of Chapter 3.

A simplified integer linear programming formulation for the case $c_v = 1$ is presented by Aardal et al. [1]. They also assume that the interference p_{vwfg} is equal for all $|f-g| \in T_{vw}$. Instead of z_{vfwg}, they introduce a new binary variable z_{vw} for every edge $\{v,w\} \in E$

$$z_{vw} = \begin{cases} 1 & \text{if the frequencies selected for } v \text{ and } w \text{ violate } T_{vw} \\ 0 & \text{otherwise} \end{cases}$$

Then, MI-FAP reads as

$$\min \quad \sum_{\{v,w\} \in E} p_{vw} z_{vw} \tag{2.33}$$

$$\text{s.t.} \quad \sum_{f \in D_v} x_{vf} = 1 \qquad \forall v \in V \tag{2.34}$$

$$x_{vf} + x_{wg} \leq 1 + z_{vw} \qquad \forall \{v,w\} \in E, f \in D_v, g \in D_w : |f-g| \in T_{vw} \tag{2.35}$$

$$x_{vf} \in \{0,1\} \qquad \forall v \in V, f \in D_v \tag{2.36}$$

$$z_{vw} \in \{0,1\} \qquad \forall \{v,w\} \in E \tag{2.37}$$

2.6.2 Benchmark Instances

In connection with the CALMA project 11 benchmark instances are available. For the instances `CELAR 09`, `CELAR 10`, `GRAPH 07`, and `GRAPH 12` not only combinations of frequencies are penalized, but also single frequency assignments. For a number of vertices,

there exists a preferred frequency f^*, which may only be deselected against a high penalty q_v. More general preference between frequencies can be modeled with penalties q_{vf} for all $v \in V$, $f \in D_v$. In that case the objective (2.33) reads

$$\sum_{\{v,w\}\in E} p_{vw} z_{vw} + \sum_{v\in V} \sum_{f\in D_v} q_{vf} x_{vf} \tag{2.38}$$

Table 2.5 shows the results of the applied methods. For the sake of completeness, the results of Chapter 4 about the tree decomposition (TD) method are included as well.

2.6.3 Lower Bounds and Exact Methods

Aardal et al. [1] applied their Branch and Cut framework for MO-FAP to solve these instances as well. Unfortunately, they were not able to solve any of these instances. For two instances they obtained a non-trivial, but poor, lower bound in this way. Tiourine, Hurkens, and Lenstra [91, 184] formulated a relaxation of the problem as a quadratic program. The quadratic program (QP) was solved by preprocessing and a branch-and-bound algorithm. For the two CELAR instances with vertex penalties q_{vf}, they succeeded to obtain fairly good lower bounds. For `CELAR 06`, De Givry, Verfaillie, and Schiex [66] proved through lower bounding techniques for constraint optimization problems that the value of the best known solution (see Section 2.6.4) is optimal. The proof of optimality was carried out by Russian Doll Search [187] on a computer network of 40 SPARC 4 workstations, and took about 3 days computation time. In Chapter 4 more optimal solutions / lower bounds are computed via a tree decomposition approach.

2.6.4 Heuristics

In Tiourine, Hurkens, and Lenstra [91, 184] also Simulated Annealing and Variable Depth Search are performed on the CELAR instances with varying success. Warners [192, 194] and Pasechnik [155] applied their potential reduction approach to MI-FAP without great success.

A standard genetic algorithm was proposed by Kapsalis, Rayward-Smith and Smith [107] (see also [106]). In Kolen [118] a genetic algorithm with optimized crossover is proposed to solve the MI-FAP. Instead of a standard crossover, the crossover routine generates the best possible child of two parents. To generate this child, we have to solve an MI-FAP with $|D_v| = 2$ for all $v \in V$. This problem can be solved to optimality with the polyhedral results of Chapter 3 (see also [121]). Applied to the instances of the CALMA project, the best known results were obtained in this way.

instance	characteristics		lower bounds				optimal	upper bounds					
	$\|V\|$	$\|E\|$	[1] LP	[184] QP	[66] CS	Ch. 4 TD	value (range)	[118] GA	[106] GA	[184] SA	[184] VDS	[194] PR	[155] PR
CELAR 06	200	1,322	5	-	3,389	3,389	3,389	3,389	3,456	3,671	3,532	4,539	4,564
CELAR 07	400	2,865	5	-	-	300,000	300,000 - 343,592	343,592	1,670,572	567,949	344,103	-	831,926
CELAR 08	916	5,744	-	-	-	150	150 - 262	262	612	276	299	-	533
CELAR 09	680	4,103	-	14,969	-	15,571	15,571	15,571	15,599	15,571	15,573	15,775	15,770
CELAR 10	680	4,103	-	31,204	-	31,516	31,516	31,516	31,517	31,516	31,516	32,460	31,517
GRAPH 05	200	1,134	-	-	-	221	221	221	293	223	-	-	452
GRAPH 06	400	2,170	-	-	-	4,123	4,123	4,123	16,020	4,189	-	-	15,047
GRAPH 07	400	2,170	-	-	-	4,324	4,324	4,324	5,990	4,324	-	-	14,183
GRAPH 11	680	3,757	-	-	-	3,016	3,016 - 3,080	3,080	30,312	3,513	-	-	14,692
GRAPH 12	680	4,017	-	-	-	11,827	11,827	11,827	15,208	11,827	-	-	17,372
GRAPH 13	916	5,273	-	-	-	9,914	9,914 - 10,110	10,110	49,205	11,130	-	-	41,784

Table 2.5: Minimum Interference benchmark instances CALMA project. Framed values indicate the optimal value.

The instances of the CALMA project are not the only problems that have inspired researchers to develop algorithms for the MI-FAP. However, only the CALMA instances can be considered as benchmark problems, since for all other sets of instances it holds that only a single group of researchers has investigated them. All research on these sets has been carried out in the direction of heuristic methods. Especially, genetic algorithms and tabu search seem to be very popular for the MI-FAP. Tabu Search is applied by Castelino, Hurley, and Stephens [33] to find an assignment with minimal unweighted interference, i.e., $p_{vw} = 1$ for all $\{v, w\} \in E$. To verify their results on large instances, they compare them with a Genetic Algorithm and a steepest descent heuristic. Computational results are reported for instances with up to 726 vertices and 75,306 edges. The number of available frequencies is 50 in all cases. In Castelino and Stephens [34, 35] tabu thresholding [72] is applied on the same instances. In [35], surrogate constraints [67] are added to the tabu thresholding approach.

Hao, Dorne and Galinier [81] also used tabu search to solve realistic instances of the French National Research Center for Telecommunications (CNET) with at most 600 transmitters. The minimum interference problem is solved as a subroutine to minimize the span of the assignment. An assignment is represented in such a way that all co-site constraints are satisfied. The length of the tabu-list is not constant, but varies during the search. Dorne and Hao [45, 46] also applied evolutionary search on a number of CNET instances with up to 300 vertices and $c_v \in \{2, 3, 4\}$. Again they would like to minimize the span of the assignment by solving repeatedly MI-FAPs. In [45] they use a mutation operator that concentrates on the change of conflicting frequencies, whereas in [46] they compare ways to deal with the co-site constraints. In [80] the same authors investigate the performance of the crossover operator in a genetic algorithm / evolutionary search.

Other genetic algorithms are given by [40, 113, 129, 150]. Crisan and Mühlenbein [40] applied a genetic algorithm to MI-FAP with tailor-made crossover and mutation operators. They solved in this way real-life instances with among 670 and 5500 transmitters. In Lai and Coghill [129], another genetic algorithm is presented to solve the minimum interference problem. Computational results are given on 2 instances. Also Ngo and Li [150] have used a genetic algorithm to solve MI-FAP. They use a special binary encoding that deals with the demand c_v for all $v \in V$, and the co-site constraints. In [174], Smith presented a genetic algorithm as well. In this case the crossover is used to reduce the adjacent and co-channel interference, whereas the mutation operator is used to reduce the co-site interference. Finally, genetic algorithms are applied by Kim et al. [113] to obtain interference free assignments. They tested several crossover and mutation operators for a couple of Philadelphia instances in which the span of available frequencies is fixed to the best lower bound of Gamst [60]. These instances were introduced by Sivrajan, McEliece and Ketchum [171]. For 5 out of the 8 instances, it is known that there exist interference-free assignments with span equal to the lower bound.

Funabiki and Takefuji [59] proposed a parallel neural network to solve the same instances. They used the Hysterses McCulloch-Pitts neuron model, instead of a Hopfield network, to solve the feasibility problem. This neural network guarantees to converge to a local optimum. With the use of some additional heuristics they hope that their approach converts to a global optimum, which is true in a substantial percentage of the cases. The first neural network approach to the feasibility FAP is due to Kunz [128], who applied a Hopfield network to the problem. Lochtie and Mehler [135, 134] also applied a neural network approach to the MI-FAP. In [135] only co-channel interference has been taken into account, whereas in [134] the results are extended to incorporate adjacent channel interference as well. Computational results are reported for a real-life 58 cell instance. Another neural network is used by Smith and Palaniswami [176]. They presented a non-linear integer programming formulation for the problem, and applied both a Hopfield and a self-organized neural network to the problem. They compared their results with simulated annealing and steepest descent on a number of Philadelphia instances by Kunz [128]. In contrast with the standard MI-FAP, the weight of the interference depends on the distance between the frequencies. The penalty is inversely proportional to the difference between the assigned frequencies.

The same cost function is applied by Young [196], who presented a local search framework. It basically consists of a frequency change neighborhood. A local search approach is also presented by Park and Lee [154]. They adjusted a local search algorithm for the k-coloring problem to the feasibility FAP. As neighborhood they apply color changing (other color for one vertex) and color interchange (interchange two colors of two vertices). Again the feasibility problem is solved as a subroutine to minimize in the end the span. Smith, Kim and Sargent [175], describe a simulated annealing approach, applied to a real life instance of a point-to-point wireless network in Jakarta, Indonesia.

Finally, Borndörfer et al. [23] extend the graph coloring heuristics of Brélaz [28] and Costa [38] to the MI-FAP to solve instances of size in between 267 and 4,240 vertices. Combined with local search, the DSATUR heuristic yield to the best solutions. They compared their algorithm with T-coloring heuristics and a heuristic based on a minimum cost flow algorithm. This last heuristic is also the topic of [24], in which the same authors present an orientation model for the FAP. This formulation forms the basis for a two stage heuristic in which an outer and inner optimization problem are solved iteratively. The outer optimization problem decides for each edge in the graph which adjacent vertex is assigned the higher frequency (orientation). The inner optimization problem is to find an assignment that respects the orientation. The inner optimization problem can be viewed as a minimum cost flow problem.

2.6.5 Conclusions

In conclusion, for the MI-FAP many heuristic procedures have been proposed by many different research groups in the context of a wide variety of applications. Among the heuristics the most promising is the Genetic Algorithm of Kolen [118], that outperforms other heuristics on the CALMA benchmark instances. A disadvantage of this technique however is that the constraint graph should not be too large or should have a low density, since otherwise the optimal crossover cannot be applied anymore [119]. For very large networks less sophisticated heuristics should be applied. Besides the proposed algorithms like tabu search, local search algorithms combined with disturbance of locally optimal solutions seems to be a promising alternative. For instance, the assignments for the instances discussed by Castelino, Hurley and Stephens [33] can be improved in this way [119].

The lack of lower bounds and exact solution techniques for the MI-FAP causes that for most methods the quality of the solutions is unknown. In view of the size of several instances, it is acceptable to suppose that no exact solution technique will ever solve these instances. However, it should be possible to derive lower bounds for these instances, and hopefully solve the smaller instances to optimality. Especially, for the benchmark instances of the CALMA project it would worthwhile to know the optimal solutions, or second best non-trivial lower bounds. The research presented in the Chapters 3 and 4 is motivated by this conclusion.

2.7 Other Models

Besides the models described in the previous subsections, other models have been proposed by several authors in the literature. We mention only a few of these approaches.

An attractive approach is to use multiple objectives that combine characteristics of the models MO-FAP, MS-FAP, MB-FAP, and MI-FAP. For example, in Duque-Antón, Kunz and Rüber [50] as well as in Al-Khaled [10] simulated annealing is applied to a FAP with cost function a linear combination of minimization of the interference, the blocking probability, and the span. Knälmann and Quellmalz [117] applied simulated annealing with cost function a convex combination of the mean interference and the maximum interference obtained by the assignment (see also Quellmalz, Knälmann and Müller [157]). An important question that remains unanswered in these approaches is the choice of the weights for the different objectives. In Walser [190] the minimum order and minimum span objectives are combined. First, the minimum order is determined. Next, the span is minimized.

Another approach is the use of n-ary constraints to model the FAP. In varying arrangements Allen, Bater, Cohen, Dunkin, and Jeavons [16, 48, 49, 103] devoted a series of papers to the fact that the use of binary constraints to model the FAP is too restrictive. They present examples in which better assignments are obtained whenever n-ary constraints, $n \geq 3$ are taken into account (see also Fischetti et al. [54]). The theory of the related constraint satisfaction problem in which n-ary constraints have to be handled as well could direct how to deal with these constraints.

In Malesińska [137, 138] and Malesińska and Panconesi [139], the combination of fixed and dynamic channel assignment is studied. They studied the case in which the cells can be partitioned in two parts, one set of transmitters that needs a fixed channel assignment scheme, and one set of transmitters that can handle dynamic channel assignment schemes. They present results on the complexity and the approximation of the performance of the dynamic part of the network, given a plan for the fixed part.

The variant of MS-FAP, in which we have a cyclic channel distance is studied by Van den Heuvel, Leese and Shepherd [85], Shepherd [167] and McDiarmid [142]. In the cyclic channel assignment problem, the frequency spectrum is supposed to be cyclic (i.e., $|f - g|_{cyclic} = \min\{|f - g|, m - |f - g|\}$, where m is the available span). For a special class of graphs, McDiarmid [142] proves that the problem can be solve in $\mathcal{O}(|V|^3)$. In [85, 167], theoretical results for infinite triangular lattices, infinite square lattices, and infinite line lattices are derived.

Finally, in Funabiki and Nishikawa [58], a FAP related to satellite communication is discussed. The problem in this case is related to the quadratic assignment problem and solved with a neural network approach. Another satellite communication problem is discussed by Thuve [181]. He modeled the problem as a set partitioning problem, and applied a heuristic solution algorithm.

2.8 Conclusions

In this survey, we have investigated the approaches proposed in recent years to solve a wide variety of frequency assignment problems. We have limited ourselves to fixed channel assignment schemes, since for these schemes it is possible to value assignments off-line by techniques from operations research and artificial intelligence. Moreover, fixed channel assignment problems can serve as bounds for the performance of dynamic and hybrid assignment schemes.

The approaches to the FAP can be classified in four categories: minimum order, minimum span, minimum blocking probability, and minimum cumulative interference. Depending on the problem, exact or heuristic methods have been proposed with varying success.

It seem that the minimum order problem can be solved nowadays quite efficiently, by combining lower bounding and heuristics. Probably the most studied FAP is the minimum span problem. Especially lower bounds have been proposed by many researchers. The best lower bounds are obtained by use of the relation with the traveling salesman problem. Heuristic methods for this problem are less far developed. At this moment no heuristics are available that solve all benchmark instances to optimality.

For the minimum blocking problem, the relation with the maximum independent set problem has directed the research to exact solution methods. Real-life instances can be solved with integer programming and efficient search techniques. Finally, the minimum interference problem is discussed. From an exact point of view, this problem seems to be the most difficult version of the FAP. Two exact methods have been applied with limited success on the benchmark instances. Most research has been carried out in the field of heuristics. Variants of genetic and tabu search algorithms have been applied to a wide variety of applications. The quality of the solutions however is still unknown, due to the lack of lower bounds.

3. The Partial Constraint Satisfaction Formulation

One of the conclusions of the previous chapter was the lack of good lower bounding techniques for the minimum interference frequency assignment problem (MI-FAP). In contrast with for instance the minimum order FAP and minimum span FAP, no combinatorial lower bounds are available for the MI-FAP. For the 11 benchmark instances available in the CALMA project only one has been solved to optimality with a very time consuming constraint satisfaction approach (cf. Section 2.6). By the $\mathcal{NP}$-hardness of the MI-FAP we cannot expect to find algorithms that solve the MI-FAP in polynomial time. Nevertheless, this chapter as well as the next chapter are devoted to exact solution methods for the MI-FAP. The goal of both chapters is to determine which exact methods can be used to solve MI-FAPs, or second best to bound the optimal value of real-life instances from below. At the same time good lower bounds / optimal solutions serve as a benchmark for the wide variety of heuristics proposed in the literature for the MI-FAP. In this chapter we formulate the MI-FAP as a partial constraint satisfaction problem with binary relations (PCSP), and analyze the problem from a polyhedral point of view.

For many combinatorial optimization problems, the most successful exact methods are based on the study of the polytope described by a (mixed) integer programming formulation (cf. Nemhauser and Wolsey [148] or Schrijver [166] for a thorough discussion of polyhedral theory, and Aardal and Van Hoesel [2, 3] for a comprehensive overview of the successful application of polyhedral combinatorics). Therefore, we study the PCSP from a polyhedral point of view.

The sequel of this chapter is organized as follows. In Section 3.1 we introduce the PCSP, describe its relation with the frequency assignment problem, and prove that the problem is $\mathcal{NP}$-hard in general. In Section 3.2 we formulate the partial constraint satisfaction problem as a binary linear programming problem, we determine the dimension of the problem, and we describe the trivial facet defining valid inequalities. The polyhedral study of the PCSP polytope is continued in Section 3.3 with the proof of two lifting theorems. The theorems are used to derive two classes of facets for the PCSP in Section 3.4. The corresponding separation problems are discussed in Section 3.5. The relation between the PCSP and the boolean quadric polytope is the topic of Section 3.6. Computational results in Section 3.7 conclude this chapter. Roughly, the Sections 3.1-3.4, and the first part of Section 3.7 are published in [121].

3.1 The partial constraint satisfaction problem

Many problems in combinatorial optimization and artificial intelligence can be modeled as *constraint satisfaction problems*. A constraint satisfaction problem (CSP) consists of (i) a set of variables, (ii) a set of possible values for each variable, the so-called domain, and (iii) a set of constraints defined on the variables. Each constraint consists of an (implicit) list of forbidden combinations of values for a set of variables. The objective is to find an assignment of values from the domains to the variables such that all constraints are satisfied. In a binary CSP every constraint restricts only combinations of values for sets of 2 variables. Alternatively, the constraints in a binary CSP can be represented by edges of a *constraint graph*, in which each vertex represents a variable. In general CSPs, the constraints can be represented by hyperedges in a hypergraph. We refer to Kumar [127] and Tsang [185] for more information about CSPs.

A CSP is called *over-constrained*, if there does not exist an assignment of values to the variables that satisfies all constraints. In this case, every solution satisfies only a number of the constraints. To make a preference among these solutions we need an objective function that assigns a value to every solution. A *partial constraint satisfaction problem* (PCSP) involves the finding of an assignment of values to variables such that a general objective function is maximized (or minimized). It is assumed that the objective function can be decomposed along the variables and constraints, i.e., the objective can be written as the cumulative of functions that involve the assignment of only one variable or two adjacent variables. In the *maximal* constraint satisfaction problem (MAX CSP) for example, the objective is to find a solution which satisfies the maximum number of constraints. We refer to Freuder and Wallace [57] for an introduction on the PCSP, and exact solution methods from artificial intelligence. In Wallace and Freuder [189] an overview of heuristic methods to solve the PCSP is given.

In this chapter we focus on the PCSP with binary constraints. The objective function penalizes both certain values, and certain binary combinations of values, and our goal is to minimize the penalty assigned to a solution. A PCSP is defined by a so-called constraint graph $G = (V, E)$. Each vertex $v \in V$ in this graph represents a decision variable, that can obtain a value from a given domain D_v. Each value has a penalty attached to it. Moreover, an edge $\{v, w\} \in E$ in the graph indicates that some combinations of domain elements of v and w are also penalized. The objective of the PCSP is to select a domain element for each vertex such that the total penalty incurred is minimized. More formally,

Partial Constraint Satisfaction
INSTANCE: Undirected graph $G = (V, E)$, finite domain set D_v for all $v \in V$, $D = \cup_{v \in V} D_v$, for all $v \in V$ vertex-penalty function $Q_v : D_v \mapsto \mathbb{Z}^+$, for all $\{v, w\} \in E$ edge-penalty

function $P_{vw} : D_v \times D_w \mapsto \mathbb{Z}^+$, and positive integer K.
QUESTION: Does there exist an assignment $(d_v)_{v \in V}$, with

$$\sum_{v \in V} Q_v(d_v) + \sum_{\{v,w\} \in E} P_{vw}(d_v, d_w) \leq K \qquad ?$$

The frequency assignment problem (FAP) in which we would like to minimize the total interference (MI-FAP) belongs to the class of PCSPs. For example, in the MI-FAP in which we have to assign a frequency to each transceiver in a mobile telephone network, a vertex corresponds to a transceiver. The domain of a vertex is the set of frequencies that can be assigned to that transceiver. An edge indicates that communication from one transceiver may interfere with communication from the other transceiver. In most applications interference occurs whenever the distance between the frequencies assigned to the transceivers is less than a given threshold depending on the two transceivers. The penalty of an edge reflects the priority with which interference should be avoided, whereas the penalty on a vertex can be seen as a level of preference for the frequencies.

The Maximum Satisfiability Problem (MAX SAT) can be reformulated as a partial constraint satisfaction problem, which implies that PCSP is NP-complete. In a MAX SAT problem m clauses $c_1, \ldots, c_m$ involving the boolean variables $x_1, \ldots, x_n$ are given. Each clause contains a number of literals, where a literal is either a variable or the negation of a variable. The problem is to assign a value true or false to each variable so as to maximize the number of clauses that are satisfied. A clause is satisfied if at least one literal in it has the value true. Formally defined,

MAXIMUM SATISFIABILITY
INSTANCE: Set $X = \{x_1, \ldots, x_n\}$ of n variables, collection $C = \{c_1, \ldots, c_m\}$ of m clauses over X, and positive integer $k \leq |C|$.
QUESTION: Is there a truth assignment for X that simultaneously satisfies at least k of the clauses in C ?

To model MAX SAT as a PCSP, we introduce a vertex v_{c_i} for every clause c_i, $i = 1, \ldots, m$, and a vertex v_{x_j} for every variable x_j, $j = 1, \ldots, n$. The domain of v_{c_i} contains an element for each literal in the clause c_i; let us denote this element by the literal itself. The domain of v_{x_j} is given by $\{true, false\}$. There is an edge between a vertex v_{c_i} representing clause c_i, and a vertex v_{x_j} representing variable x_j if and only if $x_j \in c_i$ or $\bar{x}_j \in c_i$ ($\bar{x}_j$ is the negation of x_j). If $x_j \in c_i$, then the penalty of the combination of domain values $(x_j, false)$ is equal to 1. If $\bar{x}_j \in c_i$, then the penalty of the combination of domain values $(\bar{x}_j, true)$ is equal to 1. All other penalties are zero. The optimal value of this partial constraint satisfaction problem is K if and only if the optimal value of the corresponding MAX SAT is $k := m - K$. Furthermore, an optimal solution of the MAX SAT is given by the domain values selected for the vertices corresponding to the variables in the optimal

solution of the partial constraint satisfaction problem. This shows that the two problems are equivalent. As a consequence, we can state the following theorem:

Theorem 3.1
Partial Constraint Satisfaction *is NP-complete.*

Since MAX 2 SAT (each clause contains at most 2 literals) is $\mathcal{NP}$-hard (Garey, Johnson and Stockmeyer [63]) a partial constraint satisfaction problem with $|D_v| = 2$ for all $v \in V$ is already $\mathcal{NP}$-hard.

Corollary 3.2
Partial Constraint Satisfaction *with* $|D_v| = 2$ *for all* $v \in V$ *is NP-complete.*

For the MAX 2 SAT problem a more compact PCSP formulation is possible. We have a vertex v_{x_j} corresponding to every variable x_j, and the domain is given by $\{true, false\}$. There is an edge $\{v_{x_i}, v_{x_j}\}$ if and only if there exists a clause containing a literal corresponding to x_i and a literal corresponding to x_j. The penalty corresponding to a combination of values for the variables x_i and x_j is equal to the number of clauses containing literals corresponding to both variables for which the given combination does not satisfy the clause.

The satisfiability problem (SAT), in which the question is whether there is an assignment of the variables for which all clauses are satisfied, can also be formulated as a partial constraint satisfaction problem as follows. There is one vertex for every clause and an edge if the two corresponding clauses contain a conflicting literal corresponding to the same variable. A combination $\{x_i, \overline{x}_i\}$ with $x_i \in C_j$ and $\overline{x}_i \in C_k$ has penalty one. All combinations corresponding to nonconflicting literals have penalty zero. A problem instance is satisfiable if and only if the corresponding partial constraint satisfaction problem instance has optimal value zero.

The PCSP can be viewed as a linearization of the boolean quadric polytope (cf. Padberg [152]) and is related to the transitive packing polytope (cf. Müller and Schulz [146]) as well. Section 3.6 is devoted to the relation between the PCSP and the boolean quadric polytope.

3.2 Formulation, Dimension and Trivial Facets

To formulate the partial constraint satisfaction problem as a $\{0,1\}$-programming problem we introduce the following binary variables for all $v \in V$, $d_v \in D_v$

$$y(v, d_v) = \begin{cases} 1 & \text{if } d_v \in D_v \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$

and for all $\{v,w\} \in E$, $d_v \in D_v$, $d_w \in D_w$

$$z(v,d_v,w,d_w) = \begin{cases} 1 & \text{if } (d_v,d_w) \in D_v \times D_w \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$$

In the sequel, let $q(v,d_v)$ and $p(v,d_v,w,d_w)$ denote $Q_v(d_v)$ and $P_{\{v,w\}}(\{d_v,d_w\})$, respectively.

A $\{0,1\}$-programming formulation of the partial constraint satisfaction problem is given by

$$\begin{aligned}
\min \quad & \sum_{\{v,w\}\in E} \sum_{d_v \in D_v} \sum_{d_w \in D_w} p(v,d_v,w,d_w) z(v,d_v,w,d_w) \\
& + \sum_{v \in V} \sum_{d_v \in D_v} q(v,d_v) y(v,d_v) && (3.1) \\
\text{s.t.} \quad & \sum_{d_v \in D_v} y(v,d_v) = 1 && \forall v \in V && (3.2) \\
& \sum_{d_w \in D_w} z(v,d_v,w,d_w) = y(v,d_v) && \forall \{v,w\} \in E, d_v \in D_v && (3.3) \\
& y(v,d_v) \in \{0,1\} && \forall v \in V, d_v \in D_v && (3.4) \\
& z(v,d_v,w,d_w) \in \{0,1\} && \forall \{v,w\} \in E, d_v \in D_v, d_w \in D_w && (3.5)
\end{aligned}$$

Constraints (3.2) model the fact that exactly one value in the domain of a vertex should be selected. Constraints (3.3) enforce that the combination of values selected for an edge should be consistent with the values selected for the vertices of that edge.

We define the partial constraint satisfaction polytope $X(PCSP)$ to be the convex hull of all $\{0,1\}$-vectors (y,z) satisfying (3.2) and (3.3), i.e.,

$$X(PCSP) = \text{conv}\,\{(y,z) : (y,z) \text{ satisfies (3.2)-(3.5) }\}$$

Although the y-variables can be eliminated from the formulation, we believe that it is more convenient to keep them in the formulation. Note that, once the y-variables are $\{0,1\}$ the z-variables are forced to be integral.

The dimension of the partial constraint satisfaction polytope is given by Theorem 3.3.

Theorem 3.3
The dimension of $X(PCSP)$, defined by $(G=(V,E),D_V)$ is

$$\sum_{v \in V} (|D_v| - 1) + \sum_{\{v,w\} \in E} (|D_v| - 1)(|D_w| - 1) \qquad (3.6)$$

Proof. We will first prove that $X(PCSP)$ satisfies $|V| + \sum_{\{v,w\}\in E}(|D_v| + |D_w| - 1)$ (number of variables minus dimension) linearly independent equalities, which implies that (3.6) is an upper bound for the dimension. These linear independent equalities are obtained by taking the $|V|$ constraints (3.2), and for every edge $\{v, w\}$ all but one $(= \sum_{\{v,w\}\in E}(|D_v| + |D_w| - 1))$ of the constraints (3.3). Note that the constraints (3.3) for a given edge $\{v, w\}$ can be viewed as the constraints of a transportation problem with suppliers indicated by (v, d_v) with supply $y(v, d_v)$ and clients indicated by (w, d_w) with demand $y(w, d_w)$. Thus, deleting one of these constraints results in a set of linear independent equalities.

Next, we will prove that (3.6) is a lower bound for the dimension by supplying $1 + \sum_{v\in V}(|D_v| - 1) + \sum_{\{v,w\}\in E}(|D_v| - 1)(|D_w| - 1)$ affinely independent feasible solutions. Note that once the y-variables are given, the z-variables are uniquely determined by constraints (3.3). To define these solutions we arbitrarily select a value $d_v^* \in D_v$. A first solution is given by $y(v, d_v^*) = 1$ for all $v \in V$.

Next, we construct $\sum_{v\in V}(|D_v| - 1)$ solutions which differ from the first solution in only one domain element: for each $v \in V$, $d_v \in D_v \setminus \{d_v^*\}$, we define the solution $y(v, d_v) = 1$, $y(w, d_w^*) = 1$ for all $w \neq v$. Lastly, we construct $\sum_{\{v,w\}\in E}(|D_v| - 1)(|D_w| - 1)$ solutions which differ from the first solution in two domain elements of adjacent vertices: for each $\{v, w\} \in E$, $d_v \in D_v \setminus \{d_v^*\}$, and $d_w \in D_w \setminus \{d_w^*\}$, we define the solution $y(v, d_v) = y(w, d_w) = 1$ and $y(u, d_u^*) = 1$ for all $u \in V \setminus \{v, w\}$. Note that all these solutions are affinely independent. ■

The following theorem shows that many of the trivial inequalities are facet defining.

Theorem 3.4
For every $\{v, w\} \in E$, $|D_v| \geq 2$, $|D_w| \geq 2$, $d_v \in D_v$, $d_w \in D_w$ the inequality

$$z(v, d_v, w, d_w) \geq 0 \tag{3.7}$$

defines a facet for $X(PCSP)$.

Proof. Among the affinely independent solutions in the proof of the previous theorem, all solutions but one satisfy (3.7) with equality. ■

3.3 Lifting theorems

In this section we will discuss two types of lifting. Combining them enables us to lift a facet defining inequality of a particular PCSP to facet defining inequalities for an extended PCSP. First, we show that a facet defining inequality remains facet defining if the

constraint graph is extended with vertices having one domain element (see Figure 3.1). Second, we show how a facet defining inequality can be extended if the domain of a vertex is extended with copies of other domain elements (see Figure 3.2).

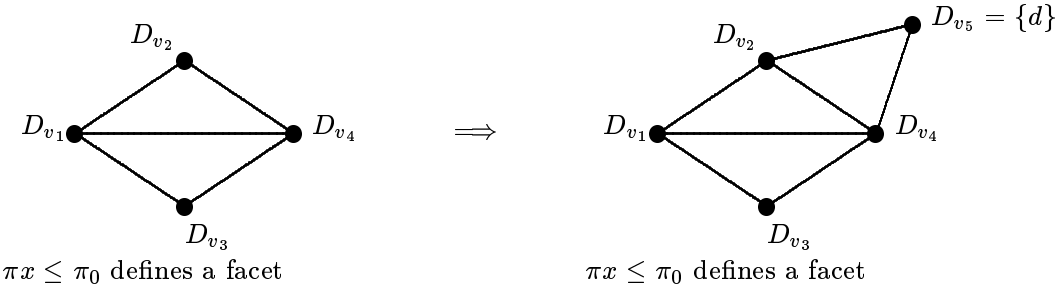


FIGURE 3.1: Extension of the graph

If $X(PCSP)$ is defined by $(G = (V, E), D_V)$, let $X_u(PCSP)$ denote the PCSP-polytope defined by the extended graph on $G_u = (V \cup \{u\}, E \cup \delta(u))$ where $\delta(u)$ is the set of edges incident to u, with the same domains for $v \in V$ and $|D_u| = 1$. Moreover, let $x = (y, z)$ denote the solution vector.

THEOREM 3.5
Let $X(PCSP)$ be defined by $(G = (V, E), D_V)$. If $\pi x \leq \pi_0$ is a facet defining inequality for $X(PCSP)$, then $\pi x \leq \pi_0$ is a facet defining inequality for $X_u(PCSP)$.

PROOF. The polytope $X(PCSP)$ is a projection of $X_u(PCSP)$ and both have the same dimension (see Theorem 3.3). ■

Next, we show how a facet defining inequality of a PCSP defined by the constraint graph $G = (V, E)$ and a set of domains D_v, $v \in V$, can be lifted into a facet defining inequality for the extended PCSP defined by the same constraint graph but different set of domains D_v^+, $v \in V$, where $D_v^+ = D_v$, for all $v \in V$, $v \neq u$, and $D_u^+ = D_u \cup \{d_u^+\}$ (see Figure 3.2). Theorem 3.8 states that if we give each variable related to d_u^+ the same coefficient as the corresponding variable of an arbitrarily selected domain element $d_u \in D_u$, then the new inequality is facet defining for the extended problem whenever the original inequality is facet defining for the original problem. In order to prove Theorem 3.8 we need the following two lemmas.

LEMMA 3.6
Let $u \in V$, $d_u \in D_u$. If

$$\beta(u, d_u)y(u, d_u) + \sum_{v \in N(u)} \sum_{d_v \in D_v} \gamma(v, d_v)z(u, d_u, v, d_v) \geq 0 \tag{3.8}$$

is a facet defining inequality for $X(PCSP)$, then the inequality describes a trivial facet.

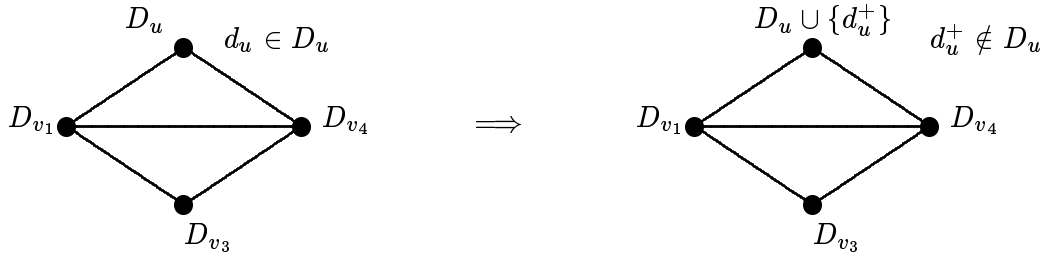


Figure 3.2: Extension of the domain

Proof. Let $d_v^* = \arg\min_{d_v \in D_v} \gamma(v, d_v)$ for all $v \in N(u)$. Adding $\gamma(v, d_v^*)$ times the model equality $y(u, d_u) - \sum_{d_v \in D_v} z(u, d_u, v, d_v) = 0$ to (3.8) for all $v \in N(u)$ results in the inequality

$$\left[\beta(u, d_u) + \sum_{v \in N(u)} \gamma(v, d_v^*)\right] y(u, d_u) + \sum_{v \in N(u)} \sum_{d_v \in D_v} [\gamma(v, d_v) - \gamma(v, d_v^*)]\, z(u, d_u, v, d_v) \geq 0$$

or using $y(u, d_u) = \sum_{d_{v'} \in D_{v'}} z(u, d_u, v', d_{v'})$ for a specific $v' \in N(u)$

$$\left[\beta(u, d_u) + \sum_{v \in N(u)} \gamma(v, d_v^*)\right] \sum_{d_{v'} \in D_{v'}} z(u, d_u, v', d_{v'}) + \sum_{v \in N(u)} \sum_{d_v \in D_v} [\gamma(v, d_v) - \gamma(v, d_v^*)]\, z(u, d_u, v, d_v) \geq 0 \tag{3.9}$$

The validity of (3.8) implies that $\beta(u, d_u) + \sum_{v \in N(u)} \gamma(v, d_v^*) \geq 0$. Thus all coefficients of (3.9) are non-negative. Furthermore, at least one coefficient is positive, otherwise (3.9) is a linear combination of the model equalities. Hence, the face defined by (3.8) is a subset of a trivial facet and thus, it can only be a trivial facet. ■

In the sequel, for a given (u, d_u), we use $x_r(u, d_u)$ and $\pi_r(u, d_u)$ as the restriction of respectively the vectors x and π to the components related to (u, d_u), i.e., the variables $y(u, d_u)$ and $z(u, d_u, v, d_v)$ for all $v \in N(u)$.

Lemma 3.7
Let $\pi x \leq \pi_0$ define a non-trivial facet of $X(PCSP)$. Then for each (u, d_u), there are exactly $1 + \sum_{v \in N(u)} (|D_v| - 1)$ solutions with $y(u, d_u) = 1$, $\pi x = \pi_0$, and for which $x_r(u, d_u)$ are affinely independent.

PROOF. Let $x^1, \ldots, x^p$ be $p = \dim X(PCSP)$ affinely independent solutions which satisfy $\pi x \leq \pi_0$ with equality. Moreover, let $x^1, \ldots, x^q$ be q solutions with $y(u, d_u) = 1$ which are affinely independent with respect to the components $y(u, d_u)$ and $z(u, d_u, v, d_v)$ for all $v \in N(u)$, $d_v \in D_v$ ($x_r^1(u, d_u), \ldots, x_r^q(u, d_u)$ are affinely independent). Then we have to prove that $q = 1 + \sum_{v \in N(u)} (|D_v| - 1)$. Since $x_r^1(u, d_u), \ldots, x_r^q(u, d_u)$ all satisfy $y(u, d_u) = 1$ these vectors are also linearly independent. So, it is sufficient to prove that the matrix $[x_r^1(u, d_u), \ldots, x_r^q(u, d_u)]$ with $1 + \sum_{v \in N(u)} |D_v|$ rows has rank $1 + \sum_{v \in N(u)} (|D_v| - 1)$. Or, equivalently, it is sufficient to prove that the dimension of the row nullspace is $|N(u)|$ (number of rows minus the rank of the matrix).

First, we prove that the dimension of the row nullspace is at least $|N(u)|$. Every solution satisfies the model equalities $y(u, d_u) - \sum_{d_v \in D_v} z(u, d_u, v, d_v) = 0$ for all $v \in N(u)$. So, if $\alpha^v = (\beta^v, \gamma^v)$ corresponds to the coefficients in the left hand side of the equality for $v \in N(u)$, then $\alpha^v x_r^i(u, d_u) = 0$ for $i = 1, \ldots, q$. Moreover, α^v, for $v \in N(u)$ are linearly independent, which implies that the dimension of the row nullspace is at least $|N(u)|$.

Now, suppose the dimension of the row nullspace is at least $|N(u)| + 1$. Then there exists another non-zero vector $\alpha = (\beta, \gamma)$ with $\alpha x_r^i(u, d_u) = 0$ for all $i = 1, \ldots, q$ which is linearly independent from the vectors α^v, $v \in N(u)$. For $j = q + 1, \ldots, p$, either $y^j(u, d_u) = z^j(u, d_u, v, d_v) = 0$ or $x_r^j(u, d_u)$ is affinely dependent of $x_r^1(u, d_u), \ldots, x_r^q(u, d_u)$. Hence, these solutions also satisfy $\alpha x_r(u, d_u) = 0$. As a consequence, the facet described by $\pi x = \pi_0$ is a subset of the face described by $\alpha x_r(u, d_u) = 0$, i.e. $F := \{x \in X(PCSP) : \pi x = \pi_0\} \subseteq \{x \in X(PCSP) : \alpha x_r(u, d_u) = 0\} =: F_\alpha$. If equality does not hold, then (since $\pi x \leq \pi_0$ describes a facet) $F_\alpha \equiv X(PCSP)$ and $\alpha x_r(u, d_u) = 0$ is an implicit equality. However, α is linearly independent from the implicit equalities involving (u, d_u). Hence $F_\alpha \equiv F$. From Nemhauser and Wolsey [148] (Theorem 3.6, page 91) it follows that either $\alpha x_r(u, d_u) \geq 0$ or $-\alpha x_r(u, d_u) \geq 0$ is a valid inequality for $X(PCSP)$ defining the same facet as $\pi x \leq \pi_0$. By Lemma 3.6, however, $\alpha x_r(u, d_u) \geq 0$ (or $-\alpha x_r(u, d_u) \geq 0$) describes a trivial facet, a contradiction. Consequently, the dimension of the row nullspace is exactly $|N(u)|$. ■

Now, we can prove the main theorem of this chapter.

THEOREM 3.8
Let $X(PCSP)$ be defined by $(G = (V, E), D_V)$. Let $u \in V$, $d_u \in D_u$. Define $X^+(PCSP)$ by $(G = (V, E), D_V^+)$ with $D_v^+ = D_v$, $v \in V \setminus \{u\}$, $D_u^+ = D_u \cup \{d_u^+\}$. If $\pi x \leq \pi_0$ is a non-trivial facet defining inequality for $X(PCSP)$, then

$$\pi x + \pi_r(u, d_u) x_r(u, d_u^+) \leq \pi_0 \tag{3.10}$$

is facet defining for $X^+(PCSP)$.

PROOF. First, note that $\dim X^+(PCSP) = \dim X(PCSP) + 1 + \sum_{v \in N(u)} (|D_v| - 1)$.

Let the solutions $x^1, \ldots, x^p$, where $p = \dim X(PCSP)$, be a set of affinely independent solutions which satisfy $\pi x \leq \pi_0$ with equality. It follows from Lemma 3.7 that there exist $1 + \sum_{v \in N(u)}(|D_v| - 1)$ solutions which satisfy $y(u, d_u) = 1$ and for which the restrictions to (u, d_u) are affinely independent. Replace in these solutions d_u by d_u^+. Then these new solutions together with the old solutions are affinely independent. ■

3.4 Non-trivial classes of facets

In this section we introduce two classes of facet defining inequalities for the PCSP. The facets are characterized by an induced subgraph $G[C] = (C, E[C])$ of the constraint graph $G = (V, E)$. For every $v \in C$ the domain D_v is partitioned into A_v and B_v. Domain values in A_v can be seen as copies of one another (i.e., their related variables have the same coefficients in the inequality); likewise the domain values in B_v. Therefore, the facet-proofs for these classes can be restricted to $G[C]$ and domains of size 2 (for all $v \in C$), which suffices according to the theorems of Section 3.3.

For notational convenience, we introduce

$$y(v, D'_v) = \sum_{d_v \in D'_v} y(v, d_v)$$

and

$$z(v, D'_v, w, D'_w) = \sum_{d_v \in D'_v} \sum_{d_w \in D'_w} z(v, d_v, w, d_w)$$

for $D'_v \subseteq D_v$ and $D'_w \subseteq D_w$.

3.4.1 The cycle inequalities

First, we introduce the *cycle inequalities*. Let the induced subgraph $G[C] = (C, E[C])$ of $G = (V, E)$ be a chordless k-cycle (i.e. $C = \{v_i : i = 1, \ldots, k\}$, $E[C] = \{\{v_i, v_{i+1}\} : i = 1, .., k-1\}\} \cup \{\{v_k, v_1\}\}$), then a k-cycle inequality, $k \geq 3$, is given by

$$\begin{aligned} \sum_{i=1}^{k-1} \big(z(v_i, A_{v_i}, v_{i+1}, A_{v_{i+1}}) + z(v_i, B_{v_i}, v_{i+1}, B_{v_{i+1}})\big) & \\ +z(v_0, A_{v_0}, v_k, B_{v_k}) + z(v_0, B_{v_0}, v_k, A_{v_k}) \qquad & \leq k - 1 \end{aligned} \tag{3.11}$$

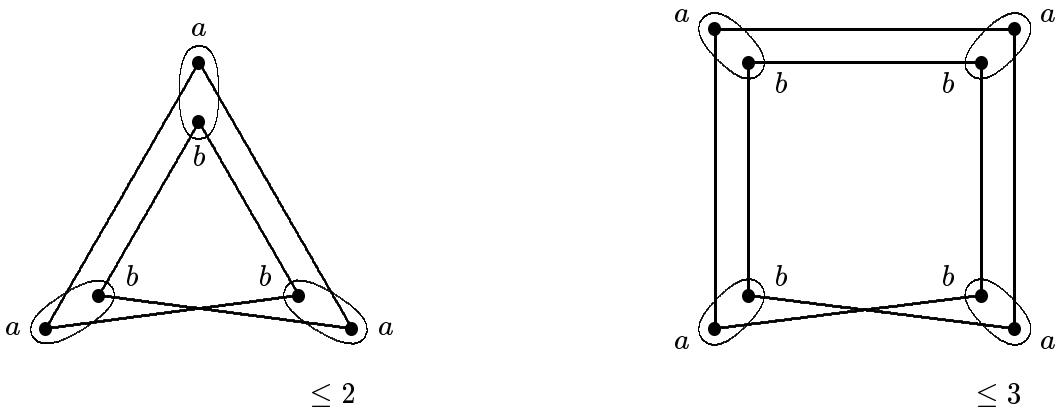


FIGURE 3.3: Cycle Inequalities

Figure 3.3 shows a 3-cycle inequality and a 4-cycle inequality. The *a-dot* represents the A-subset of the domain; the *b-dot* represents the B-subset of the domain. A line between two dots indicates that the coefficient corresponding to the indicated subsets is equal to one.

THEOREM 3.9
The k-cycle inequalities, $k \geq 3$, are valid and facet defining for $X(PCSP)$.

PROOF. By Theorem 3.5 and Theorem 3.8, it is sufficient to prove that the k-cycle inequalities are valid and facet defining for $X(PCSP)$ defined by the k-cycle constraint graph and $A_{v_i} = \{a_{v_i}\}$, $B_{v_i} = \{b_{v_i}\}$, $i = 1, \ldots, k$.

Consider an arbitrary solution x. Each edge of the cycle in the constraint graph contributes at most one to the left hand side of (3.11). So, if at least one edge does not contribute to the left hand side, (3.11) is satisfied by x. If all edges $\{v_i, v_{i+1}\}$ for $i = 1, \ldots, k-1$ contribute 1 to the left hand side, then either a_{v_i} is selected, for $i = 1, \ldots, k$ or b_{v_i} is selected, for $i = 1, \ldots, k$. But, then the edge $\{v_k, v_1\}$ does not contribute to the left hand side. Hence, x satisfies (3.11).

A k-cycle inequality is satisfied with equality if exactly one edge of the cycle does not contribute 1 to the left hand side. The k solutions ($j \in \{1, \ldots, k\}$) in which a_{v_i} is selected for $1 \leq i \leq j$ and b_{v_i} for $j+1 \leq i \leq k$ satisfy (3.11) with equality. Also, the k solutions ($j \in \{1, \ldots, k\}$) in which b_{v_i} is selected for $1 \leq i \leq j$ and a_{v_i} for $j+1 \leq i \leq k$ satisfy (3.11) with equality. These $2k = \dim X(PCSP)$ solutions are affinely independent. ■

3.4.2 The clique-cycle inequalities

A second class of facet defining valid inequalities are the *clique-cycle inequalities*. Let the induced subgraph $G[C] = (C, E[C])$ be a $k-$clique, then a $k-$clique-cycle inequality, $k \geq 3$, is defined by

$$\sum_{i=1}^{k} z(v_i, A_{v_i}, v_{i+1}, D_{v_{i+1}}) + \sum_{i<j} z(v_i, B_{v_i}, v_j, B_{v_j}) \geq k - 1 \tag{3.12}$$

with $k + 1 \equiv 1$. Note that, the inequality (3.12) is equivalent to

$$\sum_{v \in C} y(v, A_v) + \sum_{\{v,w\} \in E[C]} z(v, B_v, w, B_w) \geq k - 1$$

Figure 3.4 shows clique-cycle inequalities for $k = 3$ and $k = 4$.

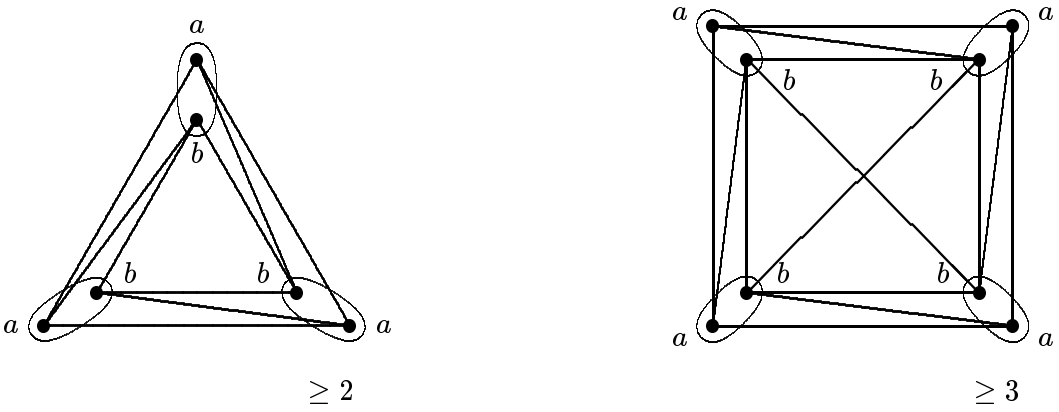


Figure 3.4: Clique-Cycle Inequalities

It should be noted that for a subset of 3 vertices of the constraint graph the clique-cycle inequality and the cycle inequality describe the same facet.

Theorem 3.10
The k-clique-cycle inequalities, $k \geq 3$, are valid and facet defining for $X(PCSP)$.

Proof. By Theorem 3.5 and Theorem 3.8, it is sufficient to prove that the k-clique-cycle inequalities are facet defining for $X(PCSP)$ defined by the k-clique constraint graph and $A_{v_i} = \{a_{v_i}\}$, $B_{v_i} = \{b_{v_i}\}$, $i = 1, \ldots, k$.

Consider an arbitrary solution x. Whenever a_{v_i} is selected for some i, then the edge $\{v_i, v_{i+1}\}$ (or $\{v_k, v_1\}$ whenever $i = k$) contributes exactly one to the left hand side

of (3.12), independent of the element selected for v_{i+1}. If both b_{v_i} and b_{v_j} are selected, then the edge $\{v_i, v_j\}$ contributes exactly one to the left hand side of (3.12). Hence, if b_v is selected for p vertices (and consequently a_v is selected for $k-p$ vertices), the total contribution to the left hand side is $\binom{p}{2} + (k-p) \geq k-1$ for all integer p.

A clique-cycle inequality is satisfied with equality, if b_v is selected for either 1 or 2 vertices. These $\binom{k}{1} + \binom{k}{2} = k + \frac{1}{2}k(k-1) = \dim X(PCSP)$ solutions are affinely independent. ■

We can extend this result to (γ, k)-clique-cycle inequalities. Let the induced subgraph $G[S]$ be a k-clique, then a (γ, k)-clique-cycle inequality, $k \geq 3$, $1 \leq \gamma \leq k-1$ is defined by

$$\gamma \sum_{v \in C} y(v, A_v) + \sum_{\{v,w\} \in E[C]} z(v, B_v, w, B_w) \geq \gamma k - \tfrac{1}{2}\gamma(\gamma+1) \tag{3.13}$$

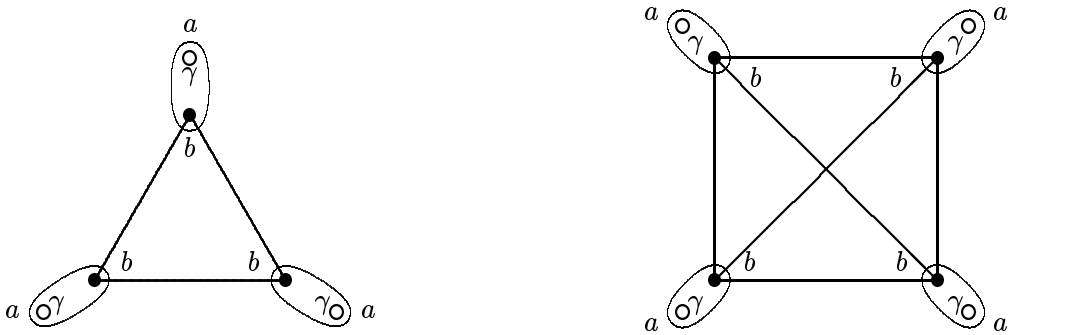


FIGURE 3.5: (γ, k)-Clique-Cycle Inequalities

For $\gamma = 1$, inequality (3.13) is equivalent with (3.12). Figure 3.5 shows (γ, k)-clique-cycle inequalities for $k = 3$, and $k = 4$.

THEOREM 3.11
The (γ, k)-clique-cycle inequalities, $k \geq 3$, are valid for $1 \leq \gamma \leq k-1$, and facet defining for $1 \leq \gamma \leq k-2$ for $X(PCSP)$.

PROOF. First, we prove validity. Let C be a clique of k vertices in G, and let $1 \leq \gamma \leq k-1$. According to Theorem 3.5 and Theorem 3.8 it suffices to prove the theorem for $G = G[C]$, $A_v = \{a_v\}$, and $B_v = \{b_v\}$ for all $v \in C$. Consider a solution x of $X(PCSP)$. Let b_v be selected for p vertices, and consequently a_v for $k-p$ vertices. Then

$$\begin{aligned} \gamma \sum_{v \in C} y(v, A_v) + \sum_{\{v,w\} \in E[C]} z(v, B_v, w, B_w) &= \gamma(k-p) + \tfrac{1}{2}p(p-1) \\ &= \gamma k + \tfrac{1}{2}p^2 - (\gamma + \tfrac{1}{2})p \end{aligned} \tag{3.14}$$

The function $f(p) = \frac{1}{2}p^2 - (\gamma + \frac{1}{2})p$ attains its minimum at $p = \gamma + \frac{1}{2}$. Since p is restricted to integral values, the minimum is attained for $p \in \{\gamma, \gamma + 1\}$. Substitution of p by γ or $\gamma + 1$ in (3.14) gives the right hand side of (3.13), which proves that the (γ, k)-clique-cycle inequality is valid $X(PCSP)$.

We continue with the proof that (3.13) defines a facet. A solution satisfies (3.13) at equality if and only if $p \in \{\gamma, \gamma + 1\}$. Consider the case $\gamma = k - 1$. If $\gamma = k - 1$, then the solutions in which b_v is selected $k - 1$ or k times satisfy (3.13) at equality. Hence, in total $k + 1$ solutions satisfy (3.13) at equality, whereas the dimension of $X(PCSP)$ is $k + \frac{1}{2}k(k-1) > k + 1$ for $k \geq 3$. So, in case $\gamma = k - 1$, (3.13) does not define a facet.

For $\gamma \in \{1, \ldots, k-2\}$, we prove that the face of $X(PCSP)$ defined by (3.13) has dimension $\dim X(PCSP) - 1$ by the identification of $\dim X(PCSP) - 1$ linearly independent vectors in the face. We construct k vectors r_i $(i = 1, \ldots, k)$, and $\frac{1}{2}k(k-1) - 1$ vectors s_{ij} $(i, j = 1, \ldots, k,\ j > i,$ and $(i, j) \neq (1, 2))$ that (i) are linearly independent, and (ii) are in the face of $X(PCSP)$ defined by the (γ, k)-clique-cycle inequality (3.13).

The vectors r_i (and s_{ij}) are defined by the components $r_i(v, d_v)$ corresponding to the $y(v, d_v)$ variables, and the components $r_i(v, d_v, w, d_w)$ corresponding to the $z(v, d_v, w, d_w)$ variables. Let $C = \{v_1, \ldots, v_k\}$ be the vertices that define the clique. For $i \in \{1, \ldots, k\}$, let r_i be a vector with the properties that

$$r_i(v_j, b_{v_j}) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases} \tag{3.15}$$

For $i, j = 1, \ldots, k,\ j > i,\ (i, j) \neq (1, 2)$, let s_{ij} be a vector with the properties that

$$s_{ij}(v, b_v) = 0 \qquad \forall v \in C \tag{3.16}$$

$$s_{ij}(v_l, b_{v_l}, v_m, b_{v_m}) = \begin{cases} 1 & \text{if } (l, m) = (1, 2), \\ -1 & \text{if } (l, m) = (i, j), \text{ and} \\ 0 & \text{otherwise} \end{cases} \tag{3.17}$$

The vectors r_i and s_{ij} are linearly independent since (a) each vector r_i has an unique non-zero element (i.e., $r_i(v_i, b_{v_i})$), and (b) among all s_{ij}, each vector s_{ij} has an unique non-zero element (i.e., $s_{ij}(v_i, b_{v_i}, v_j, b_{v_j})$).

To prove that r_i and s_{ij} are vectors in the hyperplane defined by $X(PCSP)$ and (3.13), we construct them through the subtraction of solutions that satisfy (3.13) at equality.

Let $V_b(x)$ be the subset of V for which b_v is selected in the solution x, i.e., $V_b(x) = \{v \in V : y(v, b_v) = 1\}$. For $i = 1, \ldots, k$, the vector r_i is constructed by taking the difference of two solutions x_{i0} and x_{i1}. Let x_{i0} be an arbitrary solution with $|V_b(x_{i0})| = \gamma$ and

$v_i \notin V_b(x_{i0})$. Now, let x_{i1} be defined by $V_b(x_{i1}) = V_b(x_{i0}) \cup \{v_i\}$. Then $r_i = x_{i1} - x_{i0}$ satisfies the properties (3.15).

Moreover, for $i, j, l \in \{1, \ldots, k\}$, let $x_{ij0}, x_{il0}, x_{ij1}, x_{il1}$ be 4 solutions that satisfy (3.13) at equality. Choose arbitrarily $\gamma - 1$ vertices S from $V \setminus \{v_i, v_j, v_l\}$. Define the solutions $x_{ij0}, x_{il0}, x_{ij1}, x_{il1}$ by

- $V_b(x_{ij0}) = S \cup \{v_j\}$,
- $V_b(x_{ij1}) = S \cup \{v_i, v_j\} = V_b(x_{ij0}) \cup \{v_i\}$,
- $V_b(x_{il0}) = S \cup \{v_l\}$, and
- $V_b(x_{il1}) = S \cup \{v_i, v_l\} = V_b(x_{il0}) \cup \{v_i\}$.

Clearly, all solutions have γ or $\gamma+1$ vertices for which b_v is selected, and thus satisfy (3.13) at equality. Now, let $\bar{r}_{ijl} = (x_{ij1} - x_{ij0}) - (x_{il1} - x_{il0})$, then $\bar{r}_{ijl}$ is a vector in the hyperplane. Finally, $s_{ij} = \bar{r}_{1i2} - \bar{r}_{ij1}$, satisfies the properties (3.16)-(3.17). ■

3.5 Separation of non-trivial facets

In the previous section we introduced two classes of facet defining valid inequalities. The accompanying separation problems are the topic of this section. We focus on the complexity of the separation problems that have to be resolved within a cutting plane approach. The separation problem for a class of valid inequalities $\mathcal{F}$ can be stated as follows: Given a vector $\tilde{x}$ either find an inequality in $\mathcal{F}$ that is violated by $\tilde{x}$, or conclude that all inequalities in $\mathcal{F}$ are satisfied. Within a cutting plane approach, the vector $\tilde{x}$ is a solution of the linear programming relaxation of (3.1)-(3.5) (completed with already added inequalities). We formulate the separation problem for both classes of valid inequalities, and discuss the question whether these decision problems can be solved in polynomial time or not.

Compared with other decision problems, the input of a separation problem is restricted. Since $\tilde{x}$ is an LP solution that satisfies certain (in)equalities the input of the separation problem contains in many cases some special structure, which probably can be used to solve the separation problem. Therefore, from the formulation of a separation problem as an $\mathcal{NP}$-hard problem we cannot conclude automatically that the separation is $\mathcal{NP}$-hard. To guarantee that the separation problem itself is $\mathcal{NP}$-hard given an LP solution, one should really reduce an $\mathcal{NP}$-hard problem to the separation problem with the condition that the input satisfies the (in)equalities of the LP relaxation. In the literature, this topic is discussed rarely. An exception is Klabjan, Nemhauser and Tovey [116]. They prove

that the separation of cover inequalities for the knapsack problem is $\mathcal{NP}$-hard if $\tilde{x}$ is an LP solution, but can be solved in polynomial time if the LP solution is an extreme point as well. Another example is the separation of cut-set inequalities for capacitated network design problems. Bienstock [19] proved that this problem is $\mathcal{NP}$-hard even if the input is restricted to a vector that satisfies all model equations (see also Brockmüller, Günlük and Wolsey [30]). A similar result is proved by Rutten [164] for the separation of the 2-partition inequalities for the clique partitioning polytope.

In the Sections 3.5.1 and 3.5.2 we discuss the complexity of the separation problems for the cycle and clique-cycle inequalities, respectively. We prove that some special case of the separation problem for cycle inequalities can be solved in polynomial time. Based on this polynomially solvable case, we propose a heuristic for the general case. The complexity of the separation problem in which the input is restricted to an LP solution remains open. For the clique-cycle inequalities we show that the separation problem can only be solved in polynomial time if either we can construct an algorithm that uses the additional information that $\tilde{x}$ is a solution of the linear programming relaxation of (3.1)-(3.5), or in case $\mathcal{P} = \mathcal{NP}$.

3.5.1 The cycle inequalities

First of all, we calculate the number of different facets obtained by the cycle inequalities (3.11).

Lemma 3.12
Let $C = \{v_1, \ldots, v_k\}$ be a chordless cycle in G. Then the number of cycle inequalities (3.11) that define different facets is $\frac{1}{2}\prod_{v \in C}(2^{|D_v|} - 2)$

Proof. A domain D_v can be partitioned in non-empty A_v and B_v in $2^{|D_v|} - 2$ ways. We have to partition the domain for every $v \in C$, which results in $\prod_{v \in C}(2^{|D_v|} - 2)$ partitions. However, given a partition A_v, B_v for all $v \in C$, the partition $\bar{A}_v := B_v$, $\bar{B}_v := A_v$ describes the same cycle-inequality (3.11). So, for a cycle C the number of different facets is given by $\frac{1}{2}\prod_{v \in C}(2^{|D_v|} - 2)$. ■

Note that, the facets defined by two cycles $C = \{v_1, \ldots, v_k\}$ and $\bar{C} = \{v_2, \ldots, v_k, v_1\}$ are the same. Consider the two cycle inequalities (3.11), defined by

(i). the chordless cycle $C = \{v_1, \ldots, v_k\}$ and the partition A_v, B_v, $v \in C$, and

(ii). the chordless cycle $\bar{C} = \{v_2, \ldots, v_k, v_1\}$, and the partition $\bar{A}_v$, $\bar{B}_v$, $v \in C$, with $\bar{A}_{v_1} := B_{v_1}$, $\bar{B}_{v_1} := A_{v_1}$, and $\bar{A}_v := A_v$, $\bar{B}_v := B_v$ for all $v \in C \setminus \{v_1\}$.

Then, the two cycle inequalities are exactly the same, and hence, they define the same facet. In other words, the starting point of the cycle does not influence the number of different cycle inequalities / facets.

From Lemma 3.12 we conclude that the number of cycle inequalities that represent different facets is exponentially large, and therefore enumeration of all these inequalities to detect violated inequalities is not possible. Before we introduce the decision problem Cycle Separation PCSP, we first state the following lemma:

Lemma 3.13
Let C be a chordless cycle, and let A_v, B_v be a partition of D_v for all $v \in C$. Moreover, let $\tilde{x}$ be a vector satisfying (3.2) and (3.3). If there is a $v \in V$ with $A_v = \emptyset$ or $B_v = \emptyset$, then the left hand side of the cycle inequality (3.11) is less than or equal to $k-1$.

Proof. Suppose there is a $v \in C$ with either $A_v = \emptyset$ or $B_v = \emptyset$. By the symmetry of the A_v and B_v subset, we may assume that $A_v = \emptyset$. Moreover, we may assume that $v \equiv v_1$. Let for $i = 1, \ldots, k$, $\alpha_i = \tilde{z}(v_i, A_{v_i}, v_{i+1}, B_{v_{i+1}})$ and $\beta_i = \tilde{z}(v_i, B_{v_i}, v_{i+1}, A_{v_{i+1}})$. Note that, $\alpha_1 = 0$, $\beta_k = 0$, and that the following equation is valid

$$\tilde{y}(v_{i+1}, A_{v_{i+1}}) = \tilde{y}(v_i, A_{v_i}) - \alpha_i + \beta_i$$

Then

$$\begin{aligned}
\omega(s) &= \sum_{i=1}^{k-1}(1 - \alpha_i - \beta_i) + \alpha_k + \beta_k \\
&= k - 1 - \sum_{i=1}^{k-1}(\alpha_i + \beta_i) + \tilde{y}(v_k, A_{v_k}) \\
&= k - 1 - \sum_{i=1}^{k-1}(\alpha_i + \beta_i) + \tilde{y}(v_{k-1}, A_{v_{k-1}}) - \alpha_{k-1} + \beta_{k-1} \\
&= k - 1 - \sum_{i=1}^{k-1}(\alpha_i + \beta_i) + \tilde{y}(v_1, A_{v_1}) - \sum_{i=1}^{k-1}(\alpha_i - \beta_i) \\
&= k - 1 - 2\sum_{i=1}^{k-1}\alpha_i \leq k - 1
\end{aligned}$$

which completes the proof. ■

So, in our search to a partition that violates the cycle inequality, we do not have to request that A_v and B_v are both non-empty. In case there exists a violated cycle inequality, the conditions that A_v and B_v are both non-empty will automatically be satisfied.

Cycle Separation PCSP

INSTANCE: Partial Constraint Satisfaction Problem (G, D, p, q). Chordless cycle $C = \{v_1, \ldots, v_k\}$ of k vertices. Fractional solution vector $\tilde{x}$.

QUESTION: Does there exist a partition of the domains D_v in A_v and B_v for all $v \in C$ such that the cycle inequality (3.11) is violated, i.e.,

$$\sum_{i=1}^{k-1} \left(\tilde{z}(v_i, A_{v_i}, v_{i+1}, A_{v_{i+1}}) + \tilde{z}(v_i, B_{v_i}, v_{i+1}, B_{v_{i+1}})\right) + \tilde{z}(v_1, A_{v_1}, v_k, B_{v_k}) + \tilde{z}(v_1, B_{v_1}, v_k, A_{v_k}) \qquad > k-1 \qquad ?$$

We model this problem as a $\{0,1\}$ quadratic programming problem. For each $v \in C$, $d_v \in D_v$ we define

$$s(v, d_v) = \begin{cases} 1 & \text{if } d_v \in B_v \\ 0 & \text{otherwise} \end{cases}$$

Then, a $\{0,1\}$ quadratic programming formulation of the cycle separation problem reads

$$\begin{aligned} \omega = \max \sum_{i=1}^{k-1} \sum_{d_{v_i} \in D_{v_i}} \sum_{d_{v_{i+1}} \in D_{v_{i+1}}} & \tilde{z}(v_i, d_{v_i}, v_{i+1}, d_{v_{i+1}}) \cdot \\ & \{s(v_i, d_{v_i}) s(v_{i+1}, d_{v_{i+1}}) + (1 - s(v_i, d_{v_i}))(1 - s(v_{i+1}, d_{v_{i+1}}))\} \\ + \sum_{d_{v_1} \in D_{v_1}} \sum_{d_{v_k} \in D_{v_k}} & \tilde{z}(v_1, d_{v_1}, v_k, d_{v_k}) \cdot \\ & \{s(v_1, d_{v_1})(1 - s(v_k, d_{v_k})) + (1 - s(v_1, d_{v_1})) s(v_k, d_{v_k})\} \\ \text{s.t.} \quad s(v, d_v) \in \{0,1\} & \qquad \forall v \in C, d_v \in D_v \end{aligned}$$

which is equivalent with the unconstrained quadratic $\{0,1\}$ programming problem

$$\omega = k - 1 + \max\{g^T s + s^T H s : s \in \{0,1\}^{\sum_{v \in C} |D_v|}\} \tag{3.18}$$

where

$$g(v, d_v) = \begin{cases} -2\tilde{y}(v, d_v) & \text{if } v = v_i,\ 2 \le i \le k-1 \\ 0 & \text{otherwise} \end{cases}$$

for all $v \in C$, $d_v \in D_v$ and

$$H(v,d_v,w,d_w)=\begin{cases} 2\tilde{z}(v,d_v,w,d_w) & \text{if } \{v,w\}=\{v_i,v_{i+1}\},\ i\in\{1,\ldots,k-1\} \\ -2\tilde{z}(v,d_v,w,d_w) & \text{if } \{v,w\}=\{v_1,v_k\} \\ 0 & \text{otherwise} \end{cases}$$

for all $\{v,w\} \in E[C]$, $d_v \in D_v$, $d_w \in D_w$. If $\omega > k-1$ then a violated inequality is found. If $\omega \leq k-1$, then no violated k-cycle inequality exists for the cycle C.

The problem (3.18) can be simplified by the following observations. Let $\omega(s)$ denote the value of the objective for a solution vector s.

Lemma 3.14
Let s be an arbitrary binary vector, and let $\tilde{y}(v,d_v)=0$ for some $v \in C$, $d_v \in D_v$. Define a new solution $\bar{s}$ equivalent to s except for $\bar{s}(v,d_v) := 1 - s(v,d_v)$. Then $\omega(s)=\omega(\bar{s})$.

Proof. Trivial. ■

Let $\tilde{D}_v = \{d_v \in D_v : \tilde{y}(v,d_v) > 0\}$ denote the subset of domain elements which can influence the value ω. Then, we can replace D_v by $\tilde{D}_v$ in (3.18). Combination of Lemma 3.13 and Lemma 3.14 leads to the following corollary.

Corollary 3.15
If there is a vertex $v \in C$ with $|\tilde{D}_v| = 1$, then $\omega \leq k-1$.

Proof. Follows directly from the fact that (3.11) can only be violated if for all $v \in V$ both subsets A_v and B_v are non-empty. ■

So, if the solution $\tilde{x}$ is integral for one of the vertices in the cycle, there does not exist a violated cycle inequality. In general, the unconstrained quadratic $\{0,1\}$ problem is $\mathcal{NP}$-hard [62], except for the special case in which all elements of H are non-negative (see Balinski [14], Rhys [159], Picard and Ratliff [156] or Hansen [79]) or the support graph $G(H)$ is is series-parallel (see Barahona [15]). In our case the matrix H is neither neither non-negative nor series-parallel. However, in case there is a vertex $v \in V$ for which only two $\tilde{y}$ variables are fractional, i.e., $|\tilde{D}_v| = 2$, the separation problem can be rewritten to an unconstrained $\{0,1\}$ quadratic program with non-negative matrix $\bar{H}$:

Theorem 3.16
If there is a vertex $v \in C$ with $|\tilde{D}_v| = 2$, then the problem Cycle Separation PCSP *can be solved in polynomial time.*

Proof. Without loss of generality we may assume that $D_{v_1} = \{a_{v_1}, b_{v_1}\}$. Since, the cycle-inequality is symmetric in A_v and B_v, and both A_v and B_v have to be non-empty

in case of a violated inequality, we may assume that $A_{v_1} = \{a_{v_1}\}$ and $B_{v_1} = \{b_{v_1}\}$. This implies that $s(v_1, a_{v_1}) = 1$ and $s(v_1, b_{v_1}) = 0$. The resulting $\{0,1\}$ quadratic program can be written as

$$\max\{\bar{g}^T s + s^T \bar{H} s : s \in \{0,1\}^{\sum_{v \in C \setminus \{v_1\}} |\tilde{D}_v|}\} \tag{3.19}$$

with for $v \in C \setminus \{v_1\}$, $d_v \in \tilde{D}_v$

$$\bar{g}(v, d_v) = \begin{cases} -2\tilde{y}(v, d_v) & \text{if } 3 \leq i \leq k-1 \\ -2\tilde{z}(v, d_v, v_1, b_{v_1}) & \text{if } i = 2 \\ -2\tilde{z}(v, d_v, v_1, a_{v_1}) & \text{if } i = k \end{cases}$$

and for all $\{v, w\} \in E[C]$, $d_v \in \tilde{D}_v$, $d_w \in \tilde{D}_w$

$$\bar{H}(v, d_v, w, d_w) = \begin{cases} 2\tilde{z}(v, d_v, w, d_w) & \text{if } \{v, w\} = \{v_i, v_{i+1}\},\ i \in \{2, \ldots, k-1\} \\ 0 & \text{otherwise} \end{cases}$$

So, all elements of $\bar{H}$ are non-negative, and hence (3.19) can be solved in polynomial time. ■

Corollary 3.17
If for a vertex $v \in C$ the partition A_v, B_v is given, then the problem Cycle Separation PCSP *can be solved in polynomial time.*

Proof. In case the partition A_v, B_v is given for a vertex $v \in V$, the variables $s(v, d_v)$ can be replaced by $s(v, A_v) = \sum_{d_v \in A_v} s(v, d_v)$ and $s(v, B_v) = \sum_{d_v \in B_v} s(v, d_v)$. As a consequence, the problem reduces to a problem with $|\tilde{D}_v| = 2$, which can be solved in polynomial time with Theorem 3.16. ■

Corollary 3.18
If $d = \min_{v \in C} |\tilde{D}_v|$, then the problem Cycle Separation PCSP *can be solved by solving 2^{d-1} unconstrained quadratic programs with non-negative cost matrix.*

Proof. There exist 2^{d-1} non-empty partitions A_v, B_v for the vertex v that attains the minimum $\min_{v \in C} |\tilde{D}_v|$. ■

The unconstrained $\{0,1\}$ quadratic program with non-negative cost matrix can be solved via a transformation to the minimum cut problem. However, the separation problem can

also be formulated directly as a minimum cut problem. By (3.2) and (3.3) we can rewrite the cycle-inequality (3.11) to

$$\sum_{i=1}^{k-1} \big(z(v_i, A_{v_i}, v_{i+1}, B_{v_{i+1}}) + z(v_i, B_{v_i}, v_{i+1}, A_{v_{i+1}})\big) \\ +z(v_0, A_{v_0}, v_k, A_{v_k}) + z(v_0, B_{v_0}, v_k, B_{v_k}) \quad \geq \quad 1 \tag{3.20}$$

Let $\tilde{D}_{v_1} = \{a_{v_1}, b_{v_1}\}$ and let $(N, \mathcal{A})$ define a digraph. For every $d_v \in \tilde{D}_v$, $v \in C \setminus \{v_1\}$ we define a node, and for both a_{v_1} and b_{v_1} two nodes, i.e., $N = (\bigcup_{v \in C \setminus \{v_1\}} \tilde{D}_v) \cup \{a_{v_1}, \bar{a}_{v_1}, b_{v_1}, \bar{b}_{v_1}\}$. For every $\tilde{z}(v_i, d_{v_i}, v_{i+1}, d_{v_{i+1}}) > 0$, $i \in \{1, \ldots, k-1\}$, we define an arc between d_{v_i} and $d_{v_{i+1}}$ with weight $\tilde{z}(v_i, d_{v_i}, v_{i+1}, d_{v_{i+1}})$, and for every $\tilde{z}(v_k, d_{v_k}, v_1, d_{v_1}) > 0$, we define an arc between d_{v_k} and $\bar{d}_{v_1}$ with weight $\tilde{z}(v_k, d_{v_k}, v_1, d_{v_1})$ (see Figure 3.6). Then

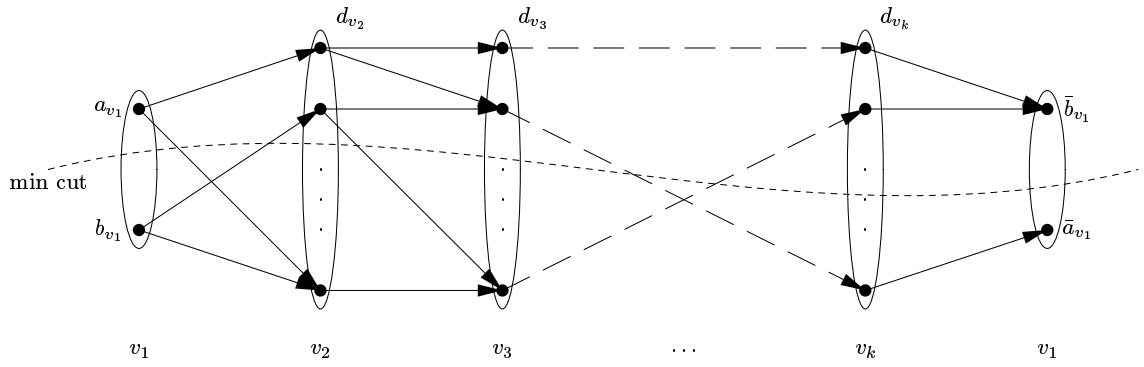


Figure 3.6: Digraph for the separation of cycle inequalities

the optimal solution of the separation problem is equivalent to a minimum weighted subset of the arcs that separates both the pairs $(a_{v_1}, \bar{a}_{v_1})$ and $(b_{v_1}, \bar{b}_{v_1})$. In case the 2-pair minimum cut is < 1 then a violated inequality is found, otherwise there does not exist a violated inequality for this cycle. Lemma 3.13 (page 65) can be helpful in this case as well. Since $|\tilde{D}_{v_1}| = 2$ each cut with value < 1 automatically separates a_{v_1} and b_{v_1} (and also $\bar{a}_{v_1}$ and $\bar{b}_{v_1}$). This implies that an 1-pair minimum cut between a_{v_1} and $\bar{a}_{v_1}$ either has value ≥ 1 or also separates b_{v_1} and $\bar{b}_{v_1}$. So, we simply have to find an 1-pair minimum cut between a_{v_1} and $\bar{a}_{v_1}$. If the minimum cut is less than 1, we have found a violated cycle-inequality, if the minimum is greater than or equal to 1, all cycle inequalities (for this cycle) are satisfied by the solution $\tilde{x}$.

If $|\tilde{D}_v| > 2$ for all $v \in C$, we can construct a digraph similar to Figure 3.6. However, in this case we have to find a minimum set of arcs that separates all pairs $\{d_{v_1}, \bar{d}_{v_1}\}$, $d_v \in \tilde{D}_v$. This problem is known as the multi-pair cut problem (see Hu [90]). The related multiterminal cut problem is discussed in Dahlhaus et al. [43]. One of their results implies that the 3-pair cut problem is $\mathcal{NP}$-hard for general graphs. Unfortunately, we cannot conclude that

Cycle Separation PCSP is $\mathcal{NP}$-hard, since the digraph has a special structure, and the weights have the property that they correspond with a solution of the LP relaxation.

In addtion to exact solution methods for the separation problem, we can also apply heuristics to the problem. Based on Corollary 3.17 and the minimum cut representation of the separation problem, we propose the following heuristic to find a violated cycle-inequality. Let v_1 be a vertex on the cycle for which $|\tilde{D}_v|$, is minimal. Construct an initial partition A_{v_1}, B_{v_1}. Given this partition for v_1, the separation problem can be solved in polynomial time, either via the unconstrained quadratic program (3.19) or via the minimum cut representation of Figure 3.6. Next, we can move a domain element $d_{v_1} \in \tilde{D}_{v_1}$ from A_{v_1} to B_{v_1} or vice versa. For the new partition we can again calculate the value of the separation problem. If the objective is improved we keep the change, otherwise we restore the original partition. The procedure can be repeated as long as there exist profitable moves.

3.5.2 The clique-cycle inequalities

Like in the previous subsection, we first state the number of different facets defined by the class of clique-cycle inequalities.

Lemma 3.19
Let C be a clique in G of $k \geq 4$ vertices, and let $\gamma \in \{1, \ldots, k-2\}$. Then the number of (γ, k)-clique-cycle inequalities (3.13) that define different facets is $\prod_{v \in C}(2^{|D_v|} - 2)$

Proof. The number of different non-empty partitions for a vertex is again $2^{|D_v|} - 2$. In contrast with the cycle inequalities each partition gives an unique inequality, which implies that the total number of different facets is equal to the total number of different partitions. ■

Note however that the (γ, k)-clique-cycle inequalities are equivalent with the $(k-1-\gamma, k)$-clique-cycle inequalities. Moreover, note that for $k = 3$ the number of different facets equals $\frac{1}{2}\prod_{v \in C}(2^{|D_v|} - 2)$, since the 3-clique is equivalent with a 3-cycle (cf. Lemma 3.12). We now define the decision problem Clique-Cycle Separation PCSP.

Clique-Cycle Separation PCSP
INSTANCE: Partial Constraint Satisfaction Problem (G, D, p, q). Clique C of $k \geq 4$ vertices. Integer $\gamma \in \{1, \ldots, k-2\}$. Fractional solution vector $\tilde{x}$.
QUESTION: Does there exist a partition of the domains D_v in A_v and B_v for all $v \in C$ such that the (γ, k)-clique-cycle inequality (3.13) is violated, i.e.,

$$\gamma \sum_{v \in C} y(v, A_v) + \sum_{\{v,w\} \in E[C]} z(v, B_v, w, B_w) < \gamma k - \tfrac{1}{2}\gamma(\gamma + 1) \qquad ?$$

Also this problem can be formulated as a $\{0,1\}$ quadratic program. Again we define variables $s(v,d_v)$ for all $v \in C$, $d_v \in D_v$

$$s(v,d_v) = \begin{cases} 1 & \text{if } d_v \in B_v \\ 0 & \text{otherwise} \end{cases}$$

Then the problem Clique-Cycle Separation PCSP reads as

$$\begin{aligned} \omega = \min \quad & \sum_{\{v,w\}\in E[C]} \sum_{d_v \in D_v} \sum_{d_w \in D_w} \tilde{z}(v,d_v,w,d_w) s(v,d_v) s(w,d_w) \\ & + \sum_{v \in C} \sum_{d_v \in D_v} \gamma \tilde{y}(v,d_v)(1 - s(v,d_v)) \\ \text{s.t.} \quad & 1 \le \sum_{d_v \in D_v} s(v,d_v) \le |D_v| - 1 && \forall v \in C \\ & s(v,d_v) \in \{0,1\} && \forall v \in C, d_v \in D_v \end{aligned}$$

which is equivalent with

$$\begin{aligned} \omega = \gamma k + \min \quad & g^T s + s^T H s && (3.21) \\ \text{s.t.} \quad & 1 \le \sum_{d_v \in D_v} s(v,d_v) \le |D_v| - 1 && \forall v \in C \quad (3.22) \\ & s(v,d_v) \in \{0,1\} && \forall v \in C, d_v \in D_v \quad (3.23) \end{aligned}$$

with for all $v \in C$, $d_v \in D_v$

$$g(v,d_v) = -\gamma \tilde{y}(v,d_v)$$

and for all $\{v,w\} \in E[C]$, $d_v \in D_v$, $d_w \in D_w$

$$H(v,d_v,w,d_w) = \tilde{z}(v,d_v,w,d_w)$$

If $\omega < \gamma k - \frac{1}{2}\gamma(\gamma+1)$ then a violated (γ,k)-clique-cycle inequality is found, else there does not exist any violated (γ,k)-clique-cycle inequality for this clique.

Lemma 3.20
Let s be an arbitrary solution that satisfies (3.22)-(3.23). Let $\tilde{y}(v,d_v) = 0$ for some $v \in C$, $d_v \in D_v$. Let $\bar{s}$ be equivalent to s except for $\bar{s}(v,d_v) = 1 - s(v,d_v)$. Then $\omega(s) = \omega(\bar{s})$.

Proof. Trivial. ■

Let $\tilde{D}_v = \{d_v \in D_v : \tilde{y}(v, d_v) > 0\}$.

Corollary 3.21
If $|\tilde{D}_v| = 2$ for all $v \in C$, then Clique-Cycle Separation PCSP *is equivalent with a PCSP defined on C with 2 domain elements.*

Proof. In case $|\tilde{D}_v| = 2$ for all $v \in C$, the constraints (3.22) reduce to constraints similar to (3.2). Linearization of the objective (3.21) gives the desired result. ■

As a consequence, if we do not have additional information on $\tilde{y}(v, d_v)$ and $\tilde{z}(v, d_v, w, d_w)$ then the problem Clique-Cycle Separation PCSP is $\mathcal{NP}$-hard. However, we know that (3.2) and (3.3) hold for every LP solution $\tilde{x}$. The question whether this additional information changes the complexity of the problem remains open. The number of clique-cycle separation problems that have to be solved can be reduced by the following lemmas.

Lemma 3.22
Let s be a $\{0,1\}$ vector. If $\sum_{d_u \in \tilde{D}_u} s(u, d_u) = 0$ for some $u \in C$, then $w(s) < \gamma k - \frac{1}{2}\gamma(\gamma+1)$ if and only if there exists a violated $(\gamma, k-1)$-clique-cycle inequality for the $k-1$ clique $C \setminus \{u\}$.

Proof. If $\sum_{d_u \in \tilde{D}_u} s(u, d_u) = 0$, then $\tilde{y}(u, A_u) = 1$, which implies that the (γ, k)-clique-cycle inequality (3.13) reduces to

$$\gamma \sum_{v \in C \setminus \{u\}} y(v, A_v) + \sum_{\{v,w\} \in E[C \setminus \{u\}]} z(v, B_v, w, B_w) \geq \gamma(k-1) - \tfrac{1}{2}\gamma(\gamma+1)$$

which is a $(\gamma, k-1)$-clique-cycle inequality for $C \setminus \{u\}$. ■

Lemma 3.23
Let s be a $\{0,1\}$ vector. If $\sum_{d_u \in \tilde{D}_u} s(u, d_u) = |\tilde{D}_u|$ for some $u \in C$, then $\omega(s) < \gamma k - \frac{1}{2}\gamma(\gamma+1)$ if and only if there exists a violated $(\gamma-1, k-1)$-clique-cycle inequality for the $k-1$ clique $C \setminus \{u\}$.

Proof. If $\sum_{d_u \in \tilde{D}_u} s(u, d_u) = |\tilde{D}_u|$, then $\tilde{y}(u, A_u) = 0$, and $z(u, B_u, v, B_v) = y(v, B_v)$ for all $v \in C \setminus \{u\}$. This implies that the (γ, k)-clique-cycle inequality (3.13) reduces to

$$\begin{aligned}(\gamma-1) \sum_{v \in C \setminus \{u\}} y(v, A_v) + \sum_{\{v,w\} \in E[C \setminus \{u\}]} z(v, B_v, w, B_w) \quad &\geq \\ \gamma(k-1) - \tfrac{1}{2}\gamma(\gamma+1) - (k-1) \quad &= \quad (\gamma-1)(k-1) - \tfrac{1}{2}(\gamma-1)\gamma\end{aligned}$$

which is a $(\gamma-1, k-1)$-clique-cycle inequality for $C \setminus \{u\}$. ■

Corollary 3.24
If there is a vertex $v \in C$ with $|\tilde{D}_v| = 1$, then there does not exist a proper (γ, k)-clique-cycle inequality that is violated.

Corollary 3.25
Let $\omega = \gamma k + \min\{g^T s + s^T H s : s \in \{0,1\}^{\sum_{v \in C} |\tilde{D}_v|}\}$. Then $\omega < \gamma k - \frac{1}{2}\gamma(\gamma+1)$ if and only if there exists a violated (σ, l)-clique-cycle inequality for some $S \subseteq C$, with $|S| = l$, and $\max\{1, \gamma - (k-l)\} \leq \sigma \leq \min\{\gamma, l-2\}$.

As a consequence, we only have to solve the clique-cycle separation problem for maximal cliques C. Although, the maximum clique problem is $\mathcal{NP}$-hard in general, in practice all maximal cliques in the constraint graph can be generated efficiently by enumeration schemes.

Besides the exact solution of (3.21)-(3.23), we can also use heuristics to solve the clique-cycle separation problem. A simple local search algorithm consists of an initialization of the subsets A_v, B_v for all $v \in C$, and local optimization by moves of domain elements from A_v to B_v or vice versa as long as improvements are obtained.

3.6 The Boolean Quadric Polytope and the PCSP

In this section we describe the relation between the PCSP and the boolean quadric polytope (BQP). The BQP is defined by the unconstrained $\{0,1\}$ quadratic program in n variables

$$\max\{c^T x + x^T Q x : x \in \{0,1\}^n\}$$

which is already mentioned in the previous section on the separation of valid inequalities. Linearization of the quadratic terms leads to a formulation with both x_i variables and $y_{ij} = x_i x_j$ variables

$$\begin{aligned}
\max \quad & \textstyle\sum_i c_i x_i + \sum_{i<j} q_{ij} y_{ij} && && (3.24)\\
\text{s.t.} \quad & x_i + x_j - y_{ij} \leq 1 && \forall i < j && (3.25)\\
& -x_i + y_{ij} \leq 0 && \forall i < j && (3.26)\\
& -x_j + y_{ij} \leq 0 && \forall i < j && (3.27)\\
& x_i \in \{0,1\} && \forall i && (3.28)\\
& y_{ij} \in \{0,1\} && \forall i < j && (3.29)
\end{aligned}$$

Then, the boolean quadric polytope is defined by

$$QP = \text{conv}\,\{(x,y) : (x,y) \text{ satisfies } (3.25) - (3.29)\}$$

In case the matrix Q is sparse, many of the y_{ij} variables can be removed from the formulation. The resulting problem can be associated with a graph $G = (V, E)$ with vertices and edges that correspond to the nonzero coefficients of c and Q. Similar QP^G is defined by

$$QP^G = \text{conv}\left\{(x,y) \in \mathbb{R}^{|V|+|E|} : (x,y) \text{ satisfies } (3.25) - (3.29) \text{ for all } \{i,j\} \in E\right\}$$

The boolean quadric polytope is studied from a polyhedral point of view by Padberg [152]. Proposition 3.26 state the relation between the PCSP and the QP^G.

Proposition 3.26
Let $X(PCSP)$ be defined by (G, D, p, q) with $D_v = \{d_v^1, d_v^2\}$ for all $v \in V$. Then there is a one-to-one correspondence between $X(PCSP)$ and QP^G.

Proof. Let $x = (y, z) \in X(PCSP)$. Define $(\bar{y}, \bar{z}) \in \mathbb{R}^{|V|+|E|}$ as

$$\bar{y}_v = y(v, d_v^2) \qquad \text{and} \quad \bar{z}_{vw} = z(v, d_v^2, w, d_w^2)$$

then $(\bar{y}, \bar{z}) \in QP^G$. On the other hand, if $(\bar{y}, \bar{z}) \in QP^G$, then $(y, z) \in \mathbb{R}^{2|V|+4|E|}$ with

$$\begin{aligned}
y(v, d_v^1) &= 1 - \bar{y}_v & y(v, d_v^2) &= \bar{y}_v \\
z(v, d_v^1, w, d_w^1) &= 1 + \bar{z}_{vw} - \bar{y}_v - \bar{y}_w & z(v, d_v^1, w, d_w^2) &= \bar{y}_w - \bar{z}_{vw} \\
z(v, d_v^2, w, d_w^1) &= \bar{y}_v - \bar{z}_{vw} & z(v, d_v^2, w, d_w^2) &= \bar{z}_{vw}
\end{aligned}$$

is an element of $X(PCSP)$. ■

Corollary 3.27
Let $X(PCSP)$ be defined by (G, D, p, q) with $D_v = \{d_v^1, d_v^2\}$ for all $v \in V$. Let $\bar{a}\bar{x} \leq \bar{a}_0$ be a facet defining inequality for the related QP^G. Then $ax \leq a_0$ is a facet defining inequality for $X(PCSP)$, with

$$\begin{aligned}
a(v, d_v^2) &= \bar{a}_v & a(v, d_v^2, w, d_w^2) &= \bar{a}_{vw} \\
a(v, d_v^1) &= a(v, d_v^1, w, d_w^1) & a(v, d_v^1, w, d_w^2) &= a(v, d_v^2, w, d_w^1) = 0
\end{aligned}$$

So, every facet defining inequality for QP^G can be transformed to a facet defining inequality for the PCSP with 2 elements per domain. With Theorem 3.8 (page 57) these inequalities can be extended to general domains. We do not have to use Theorem 3.5, since Padberg [152] proved a similar theorem about the extension of the graph. In the same paper several classes of facet defining inequalities are presented.

Proposition 3.28 ([152], Proposition 3, 4 and 5)
The inequalities (3.25)-(3.27) for $\{i,j\} \in E$ define facets of QP^G.

The transformation of these inequalities to the PCSP results in $z(v,d_v,w,d_w) \geq 0$, the trivial facets of $X(PCSP)$.

Theorem 3.29 ([152], Lemma 2 and Theorem 4)
Let C be a clique in G, $|C| \geq 2$, and let γ be an integer. The clique-inequality

$$\gamma \sum_{v \in C} x_v - \sum_{\{v,w\} \in E[C]} y_{vw} \leq \tfrac{1}{2}\gamma(\gamma+1) \tag{3.30}$$

is valid for $\gamma \in \{1,\ldots,|C|-1\}$ and defines a facet of QP^G for $\gamma \in \{1,\ldots,|C|-2\}$.

Transformation of (3.30) leads to a $(\gamma,|C|)$-clique-cycle inequality (3.13).

Theorem 3.30 ([152], Lemma 3 and Theorem 5)
Let $S \cup T$ be a clique in G with $|S| \geq 1$, $|T| \geq 2$. Then the cut inequality

$$-\sum_{v \in S} x_i - \sum_{\{v,w\} \in E[S]} y_{vw} + \sum_{\{v,w\} \in \delta(S,T)} y_{vw} - \sum_{\{v,w\} \in E[T]} y_{vw} \leq 0 \tag{3.31}$$

is valid and facet defining for QP^G.

Transformation of these inequalities and application of Theorem 3.8 results in a new class of valid inequalities for the PCSP, the *cut inequalities*.

Corollary 3.31
Let $S \cup T$ be a clique in G with $|S| \geq 1$, $|T| \geq 2$, and let A_v, B_v be a partition of D_v for all $v \in S \cup T$. Then the cut inequality

$$\begin{aligned}\sum_{\{v,w\} \in E[S]} z(v,B_v,w,B_w) + \sum_{\{v,w\} \in E[T]} z(v,B_v,w,B_w) \\ + \sum_{v \in S} y(v,B_v) \quad \geq \sum_{\{v,w\} \in \delta(S,T)} z(v,B_v,w,B_w)\end{aligned} \tag{3.32}$$

is valid and facet defining for $X(PCSP)$.

Padberg [152] generalized the cut inequality (3.31) via a symmetry argument.

Theorem 3.32 ([152], Corollary 1)
Let $S \cup T$ be a clique in G with $s = |S| \geq 1$, $t = |T| \geq 2$. Then the generalized cut inequality

$$\begin{aligned}(s-t)\sum_{v\in S} x_i + (t-s-1)\sum_{v\in T} x_v \\ - \sum_{\{v,w\}\in E[S]} y_{vw} + \sum_{\{v,w\}\in\delta(S,T)} y_{vw} - \sum_{\{v,w\}\in E[T]} y_{vw} \leq \tfrac{1}{2}(t-s)(t-s-1)\end{aligned} \quad (3.33)$$

is valid and facet defining for QP^G.

The equivalent generalized cut inequality for the PCSP is given by Corollary 3.33.

Corollary 3.33
Let $S \cup T$ be a clique in G with $s = |S| \geq 1$, $t = |T| \geq 2$, and let A_v, B_v be a partition of D_v for all $v \in S \cup T$. Then the generalized cut inequality

$$\begin{aligned}(t-s)\sum_{v\in S} y(v,B_v) + (s-t+1)\sum_{v\in T} y(v,B_v) + \sum_{\{v,w\}\in E[S]} z(v,B_v,w,B_w) \\ + \sum_{\{v,w\}\in E[T]} z(v,B_v,w,B_w) - \sum_{\{v,w\}\in\delta(S,T)} z(v,B_v,w,B_w) \geq \tfrac{1}{2}(t-s)(s-t+1)\end{aligned} \quad (3.34)$$

is valid and facet defining for $X(PCSP)$.

Concluding, the clique-cycle inequalities can be explained by a transformation of the clique inequalities of Padberg [152], and several other classes of inequalities can be generated in the same way. The cycle inequalities cannot be explained by a direct transformation of inequalities in [152] for the boolean quadric polytope to the PCSP. Even more facets for $X(PCSP)$ can be obtained via the results of Sherali, Lee and Adams [168], and De Simone [169]. Sherali, Lee and Adams [168] used a simultaneous lifting strategy to identify a class of facets that subsume the clique, cut, and generalized cut inequalities of Padberg [152]. De Simone [169] showed that the boolean quadric polytope and the cut polytope are equivalent. As a consequence, every facet of the cut-polytope can be transformed to a facet for the boolean quadric polytope, which in its turn can be transformed to a facet for $X(PCSP)$. In fact, the facets identified by Sherali, Lee and Adams [168] belong to the class of linear hypermetric inequalities of De Simone [169, 170].

3.7 Computational Results

The results of the polyhedral analysis are tested on several types of instances. First of all, a test of the quality of the valid inequalities described above is done on 11 instances with $|D_v| = 2$ for $v \in V$. These instances are subproblems of the `CELAR 08` instance of the CALMA-project (see Sections 2.2 and 2.6). For these frequency assignment problems, Kolen [118] described a genetic algorithm in which the crossover is optimized, i.e. given 2 solutions (the parents) we would like to obtain the best possible solution among all solutions that can be generated with these parents. So the crossover problem corresponds to a PCSP with at most 2 values per domain. By applying the cycle and clique-cycle inequalities these subproblems can be solved efficiently. To illustrate the efficiency of the classes of inequalities, we have selected the already mentioned 11 subproblems. We used the callable library of CPLEX 4.0 to solve the linear programming relaxation (v_{LP}), the integer programming problem (v_{IP}) as well as the linear programming relaxation with 3-cycle valid inequalities (v_3). The separation of violated valid inequalities was done by enumeration of all valid inequalities with $k = 3$ (i.e., 4 valid inequalities for each 3-cycle were available). As mentioned in Section 2.2.4 the number of vertices of the constraint graph can be reduced to half the original number, due to the equality constraints. Hence, for all instances $|V| = 458$ and $|E| = 1655$. The results are presented in Table 3.1. The program written in C++ was running on a DEC 2100 A500MP workstation with 128Mb internal memory. The table shows that for all instances the LP relaxation with 3-cycle valid inequalities gives an integer solution. The number of violated inequalities which had to be added is given in the last column. The computation times were on average reduced by 76.4%.

Another instance with a large gap between LP and IP is `p1`. This instance has 708 vertices and 1677 edges (again all domains contain 2 values). The 3-cycle inequalities close 92.6% the gap between LP and IP. With these valid inequalities CPLEX needed 113 branch-and-bound nodes to obtain and prove the optimal value. CPLEX was not able to solve this instance to optimality without adding valid inequalities.

The separation of the cycle inequalities has been done in an exact way. We simply enumerated the 4 different 3-cycle inequalities for every 3-cycle in the graph. For problems with more than 2 elements per domain, the separation problem can be solved either via the quadratic program (3.18), or via the heuristic described in Section 3.5.1. We tested both methods on a set of 5 instances with 100 vertices, 350 edges, and 2, 3, 4, 5, or 6 elements in each domain. These instances are obtained by arbitrarily selecting a subset of the domain elements from the instance `CELAR 06` of the CALMA-project. In each iteration of the cutting plane algorithm, we added at most 1 violated cycle inequality for every 3-cycle in the graph. If no violated inequalities were found anymore, we started the branch-and-bound procedure of CPLEX. For the exact separation problems, we solved the linearization of the quadratic program (3.18) via the standard branch-and-bound routine

instance	v_{LP}	v_3	v_{IP}	CPU v_{LP}	CPU v_3	CPU v_{3+IP}	CPU v_{IP}	#v.i.
c8_1	848.5	986	986	8.8	18.1	18.1	78.0	1,104
c8_2	721	836	836	8.7	11.4	11.4	48.4	497
c8_3	630.5	747	747	7.8	13.1	13.1	63.1	771
c8_4	802	834	834	8.0	10.9	10.9	35.4	1,243
c8_5	627.5	729	729	7.5	11.3	11.3	35.7	608
c8_6	695	717	717	8.6	12.0	12.0	31.5	907
c8_7	836	894	894	8.2	9.9	9.9	39.1	267
c8_8	757	835	835	7.2	10.5	10.5	71.2	747
c8_9	769	866	866	9.2	12.6	12.6	54.9	610
c8_10	768.5	812	812	8.1	10.0	10.0	37.7	215
c8_11	622	814	814	7.3	16.0	16.0	187.1	1,259
p1	35.5	104.5	110	6.6	25.5	152.4	-	266

Table 3.1: Computational results $|D_v| = 2$

of CPLEX. Table 3.2 shows for each instance the percentage the gap between LP and IP that is closed, and the cpu-times for both the cutting plane phase (v_3) and the complete procedure (v_{3+IP}). The gap is closed with more than 90% in all cases. The difference in quality between the exact and heuristic separation is negligible. Also, the difference in speed between the algorithms does not provide us with a preference for either the exact or the heuristic way of separation. For all but the last instance the heuristic separation routine is substantially faster. However, due to the inferior lower bound (part of) the benefit is nullified in the branch-and-bound procedure that have to be applied afterwards. This is especially true for instance celar6c, where the cutting plane closed 'only' 90% of the gap. For this case, we also applied another approach, in which we separate the

instance	$\|D_v\|$	CPU	exact separation			heuristic separation		
		time	gap closed	CPU-time for		gap closed	CPU-time for	
		LP	by (%)	v_3	v_{3+IP}	by (%)	v_3	v_{3+IP}
celar6a	2	0.7	99.75	9.2	9.9	99.75	2.5	2.9
celar6b	3	1.6	98.80	196.5	216.0	98.66	34.7	41.4
celar6c	4	3.1	90.96	784.2	1,309.6	90.41	488.3	1,486.4
celar6d	5	3.9	97.78	10,550.8	13,198.0	97.35	9,308.3	12,149.0
celar6e	6	4.5	97.00	29,771.3	35,234.8	96.88	68,315.4	74,142.5

Table 3.2: Comparison of exact and heuristic separation for instances with $|D_v| \geq 2$

inequalities heuristically, except when no violated inequalities were found anymore. Then, exact separation is applied to improve the result of the cutting plane phase. In this way, we reached the same lower bound as in the exact case in only 494 seconds.

method		celar6a	celar6b	celar6c	celar6d	celar6e
	$\|D_v\|$	2	3	4	5	6
	CPU-time LP	0.7	1.6	2.9	3.8	4.4
	value LP	39,507.5	26,678	10,763.3	2,492	374
	value IP	60,342.0	45,053	30,113	13,498	11,582
3-cycles	value cuts	60,290.5	44,831.8	28,381.8	13,253.6	11,243.6
	gap closed by (%)	99.75	98.80	90.96	97.78	96.98
	CPU-time LP + vi	2.8	61.2	471.2	21,120.1	89,568.2
	CPU-time IP	3.2	67.7	1,390.0	22,833.0	96,298.9
	# sep. rounds	1	21	31	827	748
	# B&B nodes	4	13	180	22	50
3+4-cycles	value cuts	60,290.5	44,913.3	28,836.8	13,254.4	11,528.3
	gap closed by (%)	99.75	99.24	93.31	97.79	99.52
	CPU-time LP + vi	2.8	42.9	633.7	23,422.9	78,440.6
	CPU-time IP	3.2	52.3	1,244.5	24,958.4	84,893.3
	# sep. rounds	1	16	67	768	629
	# B&B nodes	4	12	104	32	92
all	value cuts	60,342.0	45,053	30,113	13,498	11,582
	gap closed by (%)	100.00	100.00	100.00	100.00	100.00
	CPU-time LP + vi	10.6	614.0	21,044.0	163,230	67,1348
	CPU-time IP	10.6	614.0	21,045.9	163,236	67,1355
	# sep. rounds	9	70	215	274	352
	# B&B nodes	0	0	2	3	1

Table 3.3: Separation of 3-cycle, 4-cycle and clique-cycle inequalities

Up to now, we have only separated 3-cycle inequalities in our cutting plane approach, whereas other k-cycle and clique-cycle inequalities are available as well. In Table 3.3 we compare the results of separation of 3-cycle inequalities only, separation of 3-cycle and 4-cycle inequalities, and separation of 3-cycle, 4-cycle as well as maximal $(1, k)$-clique-cycle inequalities. All separation algorithms are done in a heuristic way, unless no violated inequalities are found in this way, then we applied exact separation. From other experiments we have concluded that separation of k-cycle inequalities for $k > 4$ has no added value. For the clique-cycle inequalities, we know from Lemma 3.22 that we only have to separate maximal cliques to obtain all violated clique-cycle inequalities.

Table 3.3 shows that in case of separating 3-cycle and 4-cycle inequalities as well as maximal $(1, k)$-clique-cycle inequalities the gap between LP and IP is closed completely in all cases, which grounded our separation of only $(1, k)$-clique-cycle inequalities instead of the general (γ, k)-clique-cycle inequalities. Table 3.3 also shows that although a larger part of the gap between LP and IP is closed when more valid inequalities are taken into account, the overall computation time is not reduced in this way. Especially the exact separation of clique-cycle inequalities causes a substantial increase of the computation time for the larger instances.

Finally, we tested the cycle inequalities (3.11) on a subgraph of the instance `CELAR 06`. In contrast with the previous experiments, the size of the domains is 44. The subgraph consists of 4 vertices and 6 edges. We separate cycle inequalities for all 3-cycles (i.e., 4 cycles). Table 3.4 shows the results. They indicate that in case of large domains the cycle

instance	v_{LP}	CPU v_{LP}	v_3	CPU v_3	# sep. rounds	v_{3+IP}	CPU v_{3+IP}	# B&B
`C6_v4`	0	1	300	552,303	5,337	300	552,366	1

Table 3.4: Results on a subgraph problem with $|D_v| = 44$.

inequalities (3.11) are strong enough to close the gap as well. However, the computation time involved in solving this instance is huge. This shows the impracticability to solve real-life instances with the polyhedral method.

3.8 Concluding Remarks

In this chapter we modeled the MI-FAP as a partial constraint satisfaction problem. We introduced an integer programming formulation for the PCSP, and analyzed the problem from a polyhedral point of view. Two lifting theorems made it possible to derive classes of facet-defining inequalities from a single inequality. Two classes of facet defining inequalities, the cycle inequalities and clique-cycle inequalities, were obtained in this way. Due to the relation with the boolean quadric polytope several other classes of facets could be derived. For the cycle inequalities and clique-cycle inequalities we discussed the accompanying separation problems. For a special case we could prove that the separation problem for cycle inequalities can be solved in polynomial time. Based on this result we described a heuristic for the general case. For the clique-cycle inequalities the complexity of the separation problem remains open. If we cannot use the additional information that the input is an LP solution, then the problem is $\mathcal{NP}$-hard.

Computational experiments indicated that for instances with small domains the 3-cycle and 4-cycle inequalities close the gap between LP and IP substantially. Moreover, if also

the clique-cycle inequalities are separated the gap is closed completely for our instances with small domains. The separation of these inequalities can be done in a heuristic or exact way. A combination of both strategies resulted in the best performance. For instances with large domains, the cutting plane approach did not lead to desirable results. Already for a very small graph, it is very time consuming to increase the objective value in this way. It seems that, although the inequalities are facet defining, many inequalities are necessary for an increase of the objective in case of large domains.

4. A Tree Decomposition Approach

Forced by the limited success of the polyhedral method (and other exact solution approaches) for real-life instances, we try to exploit the structure of the constraint graph more directly in this chapter. Instances of the MI-FAP have a geographical nature, since each antenna is placed in a two-dimensional map. Moreover, this geography influences the interference, since pairs of antennae have no interference if their distance is far enough. Finally, concentrations of antennae are found in densely populated areas. These areas are connected with one another with a limited number of edges. This led to believe that many instances have a constraint graph with a *tree-like structure*, and thus may be solved using a *tree decomposition* of the constraint graph with small *treewidth*. The notions treewidth and tree decomposition are introduced by Robertson and Seymour [162] in their fundamental work on graph minors. Besides the major role they play in graph theory, many $\mathcal{NP}$-hard problems on graphs have been shown to be solvable in polynomial (linear) time on graphs with bounded treewidth. For instance, for list coloring Jansen and Scheffler [94] showed that the problem can e solved in polynomial time. Bodlaender [21] presented an overview of $\mathcal{NP}$-hard problems that can be solved if the treewidth is bounded by a constant. We used this idea, together with sophisticated processing techniques, on a set of instances for which the previous techniques generated only few significant results, i.e., only for a small set of instances non-trivial lower bounds were computed (cf. Section 2.6). We are now able to solve many of these instances to optimality. Moreover, in an iterative version of our algorithm we are able to generate good lower bounds on the very difficult instances. The algorithm is applicable on many instances. The only serious limitation is the treewidth of the constraint graph. Finally, we mention that the approach is not restricted to MI-FAPs but can also applied to the more general partial constraint satisfaction problem with binary relations (PCSP).

The main purpose of this chapter is twofold. In the first place, our goal is to find benchmarks for a set of publicly available MI-FAPs. Secondly, our purpose is to show that the concept of tree decomposition is not only of theoretical value, but can really be used to solve combinatorial optimization problems to optimality. We do not have the intention to demonstrate this method as *the* method to solve MI-FAPs. For that purpose the method is not competitive compared with available heuristics.

The remainder of this chapter is organized as follows. In Section 4.1 we introduce the

graph theoretic concepts we use in the chapter, such as treewidth. For a description of the MI-FAP, we refer to Section 3.1 and Section 3.2. In Section 4.2 we describe the heuristic method we use to obtain a tree decomposition of the constraint graph, and in Section 4.3 we propose the dynamic programming algorithm based on the tree decomposition of the constraint graph. The practical utility of the algorithm can be improved via the use of (pre)processing techniques, which are described in Section 4.4. We present an iterative extension of the algorithm that provides lower bounds for the original problem in Section 4.5. The computational results obtained with these methods are the topic of Section 4.6. This chapter is based on [123]. A preliminary version is published in [122], whereas also an extended abstract appeared in [124].

In the sequel of this chapter we use the following notation. Let $N(v) = \{w \in V : \{v, w\} \in E\}$ denote the set of vertices adjacent to $v \in V$, whereas $N(S) = \{w \in V \setminus S : \exists_{v \in S}\{v, w\} \in E\}$ denotes the neighbors of the vertices in the subset $S \subseteq V$. Moreover, let $\delta(S, T)$ denote the set of all edges between the vertices in $S \subseteq V$ and $T \subseteq V$, i.e., $\delta(S, T) = \{\{v, w\} \in E : v \in S, w \in T\}$. We use $\delta(S)$ as short version of $\delta(S, V \setminus S)$. With $E[S]$ we denote all edges with both vertices in S, i.e., $E[S] = \delta(S, S)$. By $G[W] = (W, E[W])$ we denote the subgraph of $G = (V, E)$ induced by W.

4.1 Graph Theoretic Concepts

In this section we introduce the graph theoretic concepts used in our solution method. We define the notions tree decomposition and treewidth, together with some (well-known) properties of these notions. We also define the concept separating vertex set, which will be used in the heuristic to construct a tree decomposition.

Before we introduce the notion of tree decomposition of a graph we start with the simpler notion of path decomposition (Robertson and Seymour [161]). A path decomposition decomposes the graph in a sequence $i = 1, \ldots, r$ of subgraphs induced by subsets $X_i \subseteq V$. All vertices and edges have to be in at least one subgraph. Moreover, if a vertex is part of two induced subgraphs, then all the subgraphs in between these two in the sequence should also contain this vertex. Or equivalently, the subgraphs for which the vertex sets contain a certain vertex should be a subsequence of the total sequence. The width of a path decomposition is given by the maximum size of the vertex sets of the subgraphs minus one. The pathwidth of a graph G is the minimum width over all path decompositions of G. Formally,

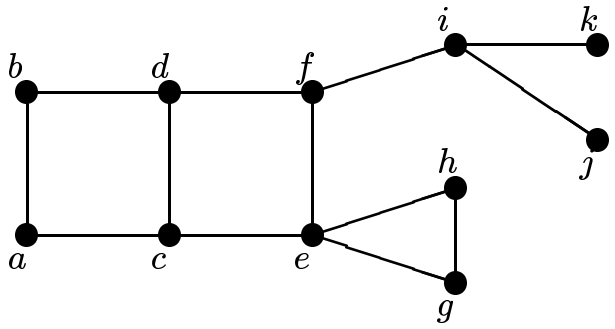
Definition 4.1 (Robertson and Seymour [161])
Let $G = (V, E)$ be a graph. A path-decomposition is a sequence $X_1, \ldots, X_r$ of subsets of V, such that

(i). $\bigcup_{i=1,\ldots,r} X_i = V$,

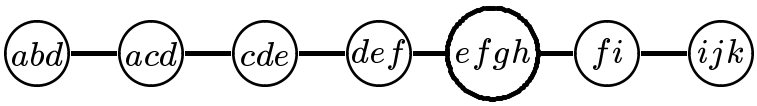
(ii). *for every edge* $\{v, w\} \in E$, *there is an* $i \in I$ *with* $v \in X_i$ *and* $w \in X_i$, *and*

(iii). *for all* $i, j, k \in \{1, \ldots, r\}$, *if* $i < j < k$, *then* $X_i \cap X_k \subseteq X_j$.

The width of a path decomposition is $max_{i=1,\ldots,r}|X_i| - 1$. *The pathwidth of a graph* G, *denoted by* $pw(G)$, *is the minimum width over all possible path decompositions of* G.



(a) graph



(b) path decomposition

Figure 4.1: Example of a path decomposition with width 3

In Figure 4.1 an example of a graph and an optimal path decomposition with width 3 is given. For special classes of graphs the pathwidth is known in advance (cf. [21]). For example, if the graph consists of a single path, the pathwidth is equal to one. For trees the pathwidth is $\mathcal{O}(d)$, where d is the length of the longest path in the tree. Robertson and Seymour developed a variant of the path decomposition concept called tree decomposition in [162]. Instead of a decomposition of the graph into a path, the graph is decomposed into a tree of induced subgraphs. The width of a tree decomposition is the maximum cardinality of the subgraphs minus one. Formally,

Definition 4.2 (Robertson and Seymour [162])
Let $G = (V, E)$ *be a graph. A tree-decomposition is a pair* $(T, \mathcal{X})$, *where* $T = (I, F)$ *is a tree with nodes* I *and edges* F, *and* $\mathcal{X} = \{X_i : i \in I\}$ *is a family of subsets of* V, *one for each node of* T, *such that*

(i). $\bigcup_{i \in I} X_i = V$,

(ii). for every edge $\{v, w\} \in E$*, there is an* $i \in I$ *with* $v \in X_i$ *and* $w \in X_i$*, and*

(iii). for all $i, j, k \in I$*, if* j *is on the path from* i *to* k *in* T*, then* $X_i \cap X_k \subseteq X_j$*.*

The width of a tree decomposition is $max_{i \in I}|X_i| - 1$*. The treewidth of a graph* G*, denoted by* $tw(G)$*, is the minimum width over all possible tree decompositions of* G*.*

The third condition of the tree decomposition is equivalent to the condition that for all $v \in V$, the set of nodes $\{i \in I : v \in X_i\}$ is connected in T. Note that, since each path decomposition is also a tree decomposition, $tw(G) \leq pw(G)$.

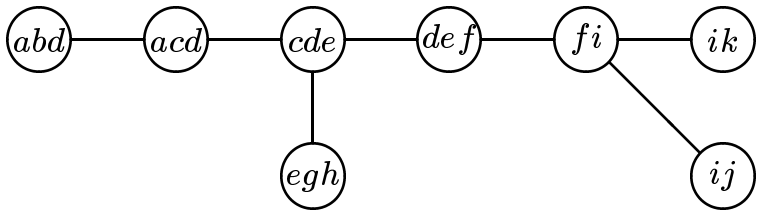


Figure 4.2: Example of a tree decomposition with width 2 for the graph of Figure 4.1(a)

In Figure 4.2 an optimal tree decomposition of the graph of Figure 4.1 is given. The width of this decomposition is 2. A connected graph has treewidth 1 if and only if the graph is a tree. The complexity of the construction of a tree decomposition (path decomposition) of minimal treewidth (pathwidth) is discussed in the next proposition.

Proposition 4.3

(i). The problem 'Given a graph $G = (V, E)$ *and an integer* k*, is the treewidth (pathwidth) of* G *at most* k*' is NP-complete.*

(ii). Given a constant integer k*, the problem 'Given a graph* $G = (V, E)$*, is the treewidth (pathwidth) of* G *at most* k*' can be solved in polynomial time.*

So, if the integer k is part of the input of the problem, the problem is NP-complete whereas it can be solved in polynomial time in case k is fixed. The NP-completeness results for treewidth and pathwidth are due to Arnborg, Corneil and Proskurowski [13]. An algorithm that solves the problem in linear time for constant k is given by Bodlaender [22]. However, this algorithm is exponential in k, and is therefore impractical for graphs with larger treewidth. Therefore, we use a heuristic to construct tree decompositions.

In a tree decomposition we can remove nodes for which the corresponding vertices form a subset of the vertices of another node. As a consequence, every tree decomposition can be

transformed to a tree decomposition in which the vertex-sets of all internal nodes separate the constraint graph in at least two components, i.e., the vertices form a separating vertex set.

Definition 4.4
An st-separating set of $G = (V, E)$ is a set $S \subseteq V \setminus \{s, t\}$ with the property that any path from s to t passes through a vertex of S. The minimal separating vertex set of G is given by the st-separating set with minimum cardinality over all combinations $\{s, t\} \notin E$.

Note that the separating vertex sets in a tree decomposition are not necessarily minimal. The property that every internal node correspond with a separating vertex set forms the basis of our heuristic, which is the topic of the next section.

4.2 Construction of a Tree-Decomposition

Since the algorithm we want to use for solving MI-FAPs heavily depends on the width of the tree decomposition of the constraint graph, we need a tree decomposition with small width. Finding a tree decomposition with optimal width is NP-hard. Therefore, we implemented a sequential improvement heuristic. The algorithm aims at decreasing the cardinality of the nodes in a given tree decomposition based on the property that the vertices that correspond to internal nodes of the tree are separating vertex sets in the graph. We try to replace a node in an existing tree decomposition by a number of new nodes for which the maximum cardinality is smaller than the cardinality of the original node. To achieve this goal, we search for small separating vertex sets. In Section 4.2.1 we describe the algorithm to find a minimum separating vertex set in a graph, whereas the heuristic itself is the topic of Section 4.2.2.

4.2.1 Minimum separating vertex set in a graph

For any combination of 2 non-adjacent vertices, the *st*-separating set with minimal cardinality can be found efficiently using Menger's theorem (see also Ahuja, Magnanti, and Orlin [9]).

Theorem 4.5 (Menger [144])
Given a graph $G = (V, E)$ and two distinct non-adjacent vertices $s, t \in V$, the minimum number of vertices in an st-separating set is equal to the maximum number of vertex-disjoint paths connecting s and t.

So, we have to calculate the maximum number of vertex-disjoint paths. This problem is solvable in polynomial time by standard network flow techniques. First, we construct a

directed graph $D = (V, A)$, in which each edge $\{v, w\}$ is replaced by two arcs (v, w) and (w, v) both with weight ∞. Next, we construct an auxiliary directed graph D' by

- replacing each vertex v by two vertices v' and v'',
- redirecting each arc with head v to v',
- redirecting each arc with tail v to v'', and
- adding an arc from v' to v'' with weight 1.

Then, the minimum number of vertices in an st-separating set in G is equal to the minimum weight of an $s'' - t'$ cut in D'. So, if we calculate the minimum $s'' - t'$ cut for every combination $s, t \in V$, $\{s, t\} \notin E$, we obtain the minimum separating vertex set. Note that since the graph D' is a directed graph, we have to solve $\mathcal{O}(n^2)$ minimum cut problems. In other words, we cannot use the algorithm of Gomory and Hu [75], which solves the all pairs minimum cut problem for undirected graphs by solving only $\mathcal{O}(n)$ minimum cut problems.

4.2.2 Heuristic

The heuristic we use to obtain a tree decomposition can be described as follows. We start with the trivial tree decomposition in which we have one node corresponding to the complete graph. During the process we have a tree decomposition $(T, \mathcal{X})$. We select the node $i \in I$ with $|X_i|$ maximum. This node is replaced by $m + 1$ nodes $i_0, \ldots, i_m$ with vertex sets $X_{i_0}, \ldots, X_{i_m}$. The nodes $i_1, \ldots, i_m$ all are connected with i_0. Each node $k \in N(i)$ is connected to exactly one node $j \in \{i_0, \ldots, i_m\}$, such that all conditions of a tree decomposition are satisfied again.

The sets $X_{i_0}, \ldots, X_{i_m}$ are defined as follows. We construct a graph $G_i = (V_i, E_i)$ that consist of the induced subgraph $G[X_i]$ and the additional edges $\cup_{k \in N(i)} C(X_i \cap X_k)$, where $C(X)$ denotes a complete graph on the vertices X (i.e., $C(X)$ is a clique). If G_i is a complete graph, then $X_{i_0} := X_i$ and $m = 0$, i.e., we do not change the tree decomposition. If G_i is not a clique, then we calculate a minimum separating vertex set $S \subset V_i$. Let $Y_{i_1}, \ldots, Y_{i_m}$ be the vertex sets of the $m \geq 2$ components of $G_i[V_i \setminus S]$. We define $X_{i_0} := S$, and $X_{i_j} := Y_{i_j} \cup S$ for all $j \in 1, \ldots, m$. The set X_k has a non-empty intersection with at most one set Y_{i_j}, $j = 1, \ldots, m$: Let $v, w \in X_i \cap X_k$, then $\{v, w\} \in C(X_i \cap X_k) \subset E_i$, which implies that v and w cannot be separated by S. So, either $v, w \in S$ or $v, w \in Y_{i_j} \cup S$ for only one $j \in \{1, \ldots, m\}$. Therefore, we connect each neighbor $k \in N(i)$ with the node i_j, $j \in \{1, \ldots, m\}$ for which the intersection of X_k and Y_{i_j} is non-empty, or in case there is none with i_0. As a consequence, the new construction is a tree again (see Figure 4.3). In the new tree the conditions for a valid tree decomposition again hold. Since

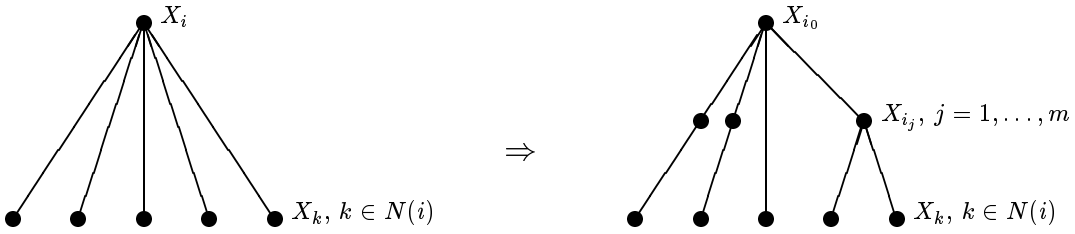


Figure 4.3: Improvement step of a tree decomposition

$\cup_{j=0}^{m} X_{i_j} = (\cup_{j=0}^{m} Y_{i_j}) \cup S = X_i$ condition (i) is satisfied. To satisfy condition (ii) we have to prove that for each edge $\{v, w\} \in E[X_i]$ one of the new vertex sets $X_{i_0}, \ldots, X_{i_m}$ contains both vertices. If $v, w \in S$, then this is trivially true. Otherwise, suppose $v \in Y_{i_j}$ for some $j \in \{1, \ldots, m\}$. If $w \in Y_{i_k}$, $k \neq j$, then S does not separate Y_{i_j} and Y_{i_k}; a contradiction. And thus, $w \in Y_{i_j} \cup S = X_{i_j}$. Condition (iii) states that all nodes in the tree that contain the same vertex v must form a subtree. We only need to check this for $v \in X_i$. If $v \in S$ then v is contained in all new nodes and the condition is trivially satisfied. Otherwise, let $v \in Y_{i_j}$ for some $j \in \{1, \ldots, m\}$. By construction, nodes $k \in N(i)$ and i_j are connected if X_k and Y_{i_j} intersect. Hence, all nodes that contain v form a subtree again.

Note that, if G_i is not a clique, then there exist vertices $v, w \in X_i$ with $\{v, w\} \notin E_i$. Thus $S = X_i \setminus \{v, w\}$ separates G_i in two components; $Y_{i_1} = \{v\}$ and $Y_{i_2} = \{w\}$. So, $\max\{|Y_{i_1} \cup S|, |Y_{i_2} \cup S|\} = |X_i| - 1 < |X_i|$. As a consequence, the width of the tree decomposition may decrease. Figure 4.4 shows the heuristic in a flowchart.

The width of the resulting tree decomposition approximates the minimal treewidth. However, as long as the separating vertex sets S form cliques in the original graph, the algorithm operates in an exact way, since the optimal tree decomposition will contain a node for every clique that separates the graph in multiple components.

4.3 Dynamic Programming Algorithm

The algorithm that solves the MI-FAP in polynomial time (given that the treewidth is at most a constant k) is based on the following idea. Let $S \subset V$ be a separating vertex set of G with $G[V \setminus S] = G[V_1] \cup G[V_2]$. Then the optimal assignment in V_1 (or V_2) only depends on the assignment in S. So, given an assignment of S the problem decomposes in two MI-FAPs on $G[V_1]$ and $G[V_2]$. Thus, the MI-FAP can be solved by solving the two MI-FAPs on $G[V_1]$ and $G[V_2]$ for all possible assignments in S. This idea can be formulated as a dynamic programming algorithm using a tree decomposition of the graph. For every internal node $i \in I$, X_i is a separating vertex set, which implies that given an assignment for X_i, the MI-FAP decomposes in smaller MI-FAPs for every branch in the tree.

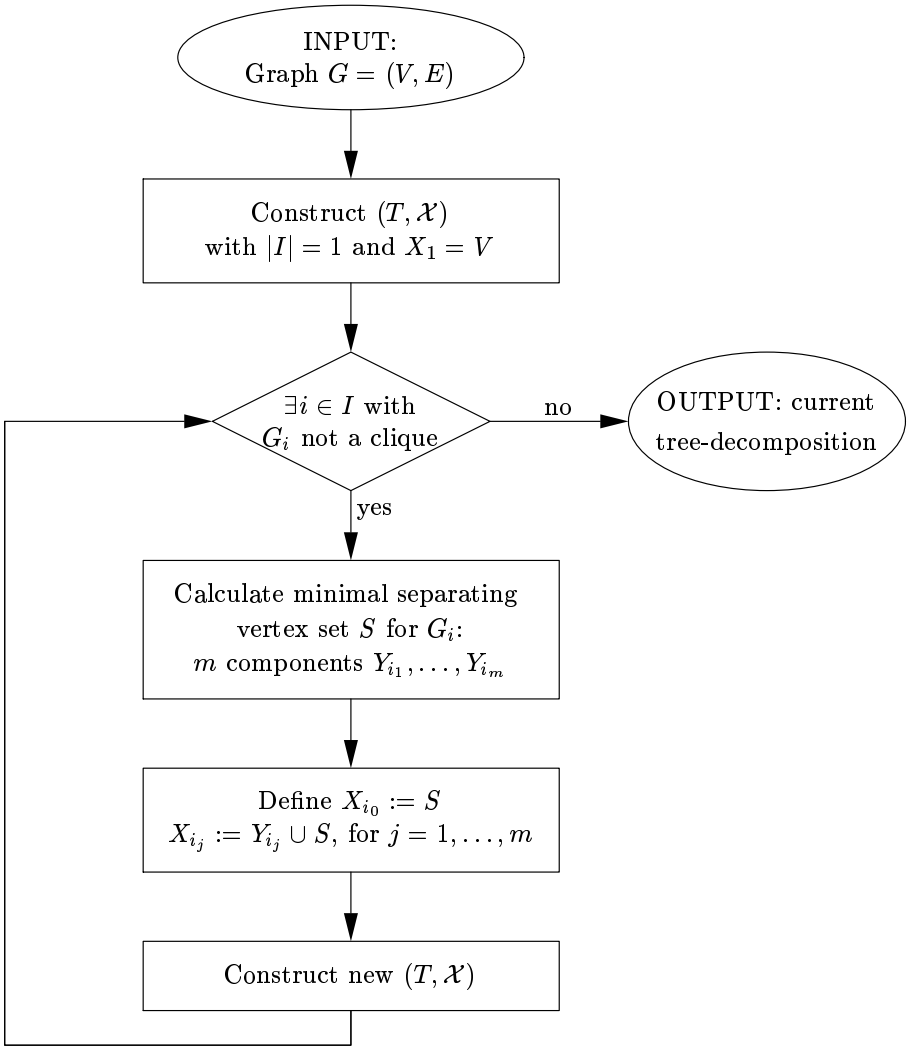


Figure 4.4: Heuristic for construction of a tree decomposition

Before we describe the algorithm in more detail, we first introduce some additional notation. In the sequel of the chapter we assume that the tree is rooted and binary. Let $Y_i = \{v \in V : \exists j \in I, j \text{ descendant of } i \text{ and } v \in X_j\}$ denote the set of vertices that is represented by the subtree rooted at node i. Given a subset $S \subseteq V$, we denote with $d_S = (d_v)_{v \in S}$ an assignment of domain elements $d_v \in D_v$ for every vertex $v \in S$. Similar, D_S denote the complete set of all assignments for a set S.

Now, we can describe the dynamic programming algorithm as follows. In a bottom-to-top way we compute for every node $i \in I$ all assignments for the subset Y_i, D_{Y_i}. Starting with a leaf $i \in I$ of the tree, the algorithm stores all assignments for the vertices in X_i. The computation of all assignments takes $\mathcal{O}(\Pi_{v \in X_i} |D_v|) = \mathcal{O}(d^{|X_i|})$ time, where $d = \max_{v \in V} |D_v|$. Next, given all assignments for two nodes $j, k \in I$ with common predecessor $i \in I$, we can compute all assignments Y_i by combining every assignment of

Y_j, every assignment of Y_k that has the same assignment for the vertices in $X_j \cap X_k$, and every assignment of domain elements to the vertices in $X_i \setminus (X_j \cup X_k)$. However, since X_i is a separating vertex set in the graph, we do not have to store all assignments for the vertices in Y_i, but only the assignments that differ for the vertices in X_i. For an assignment of the vertices in X_i, we only have to store the best assignment for the vertices in $Y_i \setminus X_i$. In other words, we have to store at most $\Pi_{v \in X_i} |D_v|$ assignments for node $i \in I$ instead of $\Pi_{v \in Y_i} |D_v|$ assignments to obtain the overall optimal solution. The computation of these assignments can be done in $\mathcal{O}(\Pi_{v \in X_i \cup X_j \cup X_k} |D_v|) = \mathcal{O}(d^{|X_i|+|X_j|+|X_k|})$. Finally, for the root node $r \in I$ of the tree T, $Y_r = V$, and so we only have to store one solution which gives the desired optimal solution for the problem. The overall computation time of this algorithm is given by $\mathcal{O}(nd^{3k})$, where k is the width of the tree decomposition $(T, \mathcal{X})$ of G that is used. So, for graphs with treewidth bounded by a constant k, this algorithm solves the MI-FAP in time polynomial in n and d, but exponential in k. In Figure 4.5 the algorithm is represented in a flowchart, where we assume that the nodes are numbered $1, \ldots, |I|$ in a topological order from top to bottom.

The performance of the algorithm highly relies on additional techniques to reduce the size of the sets of assignments D_{Y_i}. These techniques are described in the next section.

4.4 Reduction Techniques

Quick ways to remove vertices and edges from the constraint graph or to remove frequencies from the domains of the vertices may speed up any solution technique for the MI-FAP applied afterwards. Our technique for solving the MI-FAP, a dynamic programming algorithm based on the tree decomposition of G, computes all non-redundant assignments for subsets of vertices. The number of different assignments grows exponentially with the cardinality of the subset, which makes the need for good reduction techniques evident. In this section we present several such techniques. All are based on the following paradigm for extending partial feasible solutions:

A partial feasible solution can be extended to an optimal solution *only if* the extension itself is optimal with respect to the partial feasible solution. In other words, if a partial feasible solution is not extended optimally, the resulting feasible solution is certainly not optimal.

In the next subsection we use this paradigm directly to remove vertices, or replace them by edges. In Subsection 4.4.2 we present a penalty shifting procedure, which is mainly used to obtain lower bounds on the value of the instances, but can sometimes remove edges from the constraint graph as well. In Subsection 4.4.3, we present techniques to remove frequencies from the domains of vertices, and to remove non-optimal partial feasible solutions. This is done in two ways: by using upper bounding techniques, and by

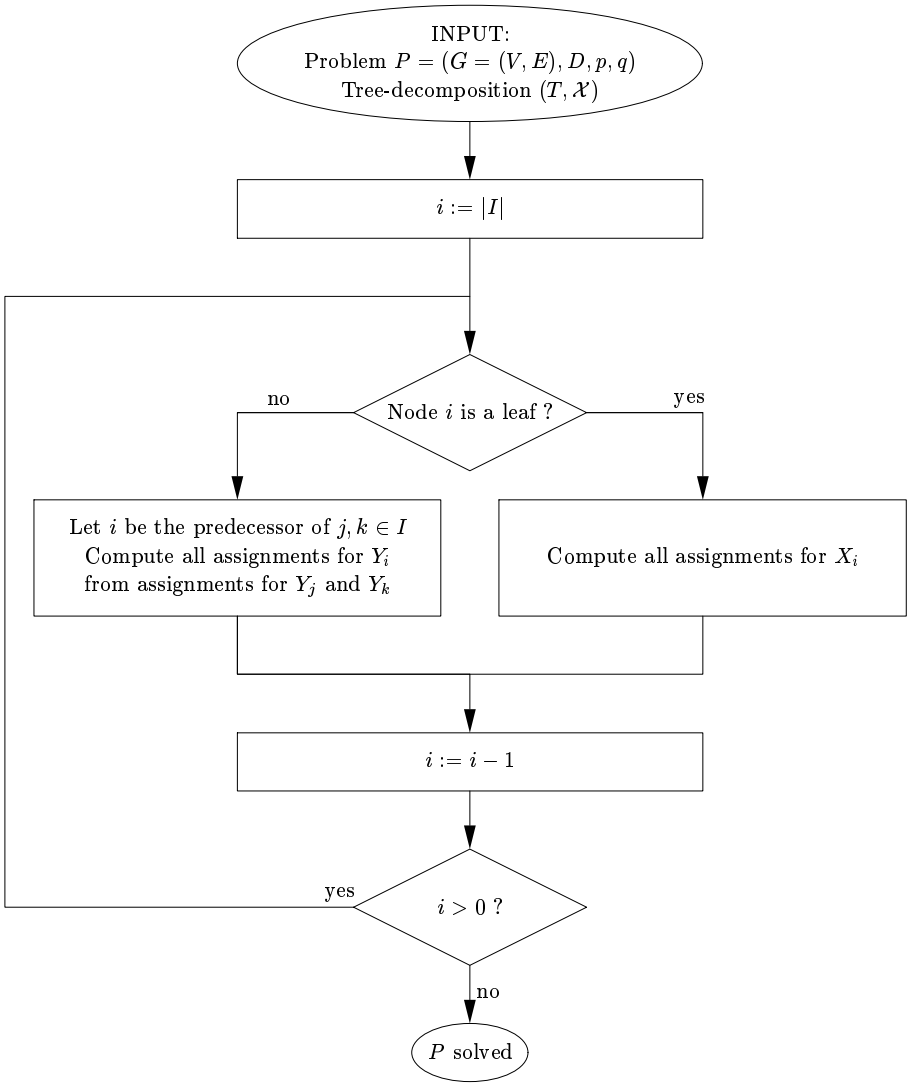


Figure 4.5: Dynamic programming algorithm

using dominance criteria.

4.4.1 Constraint graph reduction

In this subsection we describe how we can remove vertices $v \in V$ with $|D_v| = 1$ or $|N(v)| \leq 2$ from G. First of all, for vertices v with $D_v = \{d_v^*\}$ we do not have a choice for the frequency. Therefore v can be removed from the constraint graph, provided that $q(v, d_v^*)$ is added to the solution value, and that for every $w \in N(v)$, $d_w \in D_w$ the penalty $p(v, d_v^*, w, d_w)$ is added to the vertex penalty $q(w, d_w)$. Vertices with degree zero can also be removed from the constraint graph. For a vertex $v \in V$ with $|N(v)| = 0$, the optimal choice of a frequency is $\arg\min_{d_v \in D_v} q(v, d_v)$. So, the vertex can be removed from the

graph, provided that the value of the optimal solution in the remaining problem will be increased with $\min_{d_v \in D_v} q(v, d_v)$.

Next, let $v \in V$ be a vertex with $|N(v)| = 1$, and let $N(v) = \{w\}$. Consider a partial solution in which v is the only vertex without a frequency assigned to it. We should assign a frequency to v, that has minimal penalty with respect to the partial solution. To do so, we only need to consider the frequency assigned to w, say d_w^*, since the other vertices are not connected to v in G, and, thus, do not influence the penalty incurred by any choice of frequency for v. Therefore, it suffices to compute the smallest penalty incurred by the frequencies of v, i.e., $\min_{d_v \in D_v}\{q(v, d_v) + p(v, d_v, w, d_w^*)\}$, and extend the partial feasible solution with a frequency d_v^* that attains this minimum. Although d_w^* may differ among all partial solutions, we can determine the best extension of any partial feasible solution beforehand by, for all $d_w \in D_w$, computing the value

$$q'(w, d_w) = \min_{d_v \in D_v}\{q(v, d_v) + p(v, d_v, w, d_w)\}$$

and subsequently adding $q'(w, d_w)$ to $q(w, d_w)$. This, in effect, adds to each d_w the optimal choice in D_v at the beginning of the algorithm, allowing us to remove the vertex v and the edge $\{v, w\}$ from the instance. At the end of the algorithm an optimal solution found for the problem instance restricted to $G[V \setminus \{v\}]$, can then be extended by selecting the optimal choice $d_v^* \in D_v$ given the chosen frequency d_w^* of w.

We can generalize this idea to vertices with degree two as follows. Let v be such a vertex, and let $N(v) = \{u, w\}$. To extend a partial solution in which v is the only node without a frequency, we should assign a frequency to v, that is optimal with respect to the frequencies of u and w. Let d_u^* and d_w^* be the selected frequencies. We then select $d_v^* = \arg\min_{d_v \in D_v}\{p(u, d_u^*, v, d_v) + q(v, d_v) + p(v, d_v, w, d_w^*)\}$. Again, we can do this beforehand by, for all $d_u \in D_u, d_w \in D_w$, computing the value

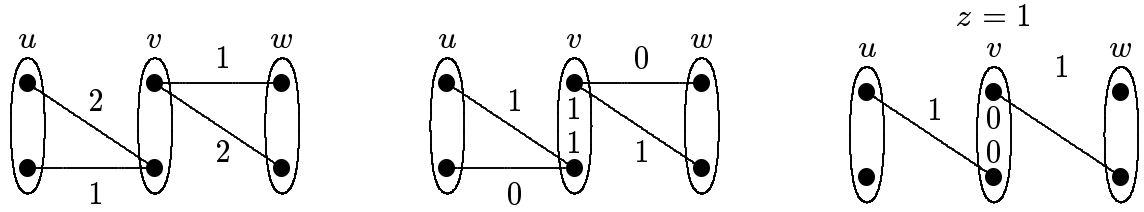
$$p'(u, d_u, w, d_w) = \min_{d_v \in D_v}\{p(u, d_u, v, d_v) + q(v, d_v) + p(v, d_v, w, d_w)\}$$

and subsequently adding $p'(u, d_u, w, d_w)$ to $p(u, d_u, w, d_w)$. This, in effect, adds to each combination $\{d_u, d_w\}$ the optimal choice in D_v, allowing us to remove the vertex v and its two incident edges from the instance. Note that possibly the edge $\{u, w\}$ may have to be inserted in the constraint graph.

We can repeat the reduction process until all vertices with degree at most two are removed.

4.4.2 Penalty shifting - Lower bounding

In this subsection we present a technique to obtain a lower bound on the optimal value of the instances by shifting penalties from edges to vertices and back, and from vertices to the objective and back. We first illustrate the technique by the example in Figure 4.6(a). We have three vertices, each with 2 domain elements. The non-zero edge-penalties are given by edges. We can transform this part of the instance by moving penalty from the penalty matrix to the penalties on frequencies (Figure 4.6(b)), and even from the frequencies to the objective (Figure 4.6(c)).



(a) original (b) shift to vertex v (c) shift to objective

Figure 4.6: Example shifting penalties

If for an edge $\{v, w\} \in E$ we have a penalty matrix such that given $d_v^* \in D_v$ for all $d_w \in D_w$, $p(v, d_v^*, w, d_w) > 0$ then by model equality (3.3) we can decrease these penalties, and simultaneously increase $q(v, d_v^*)$ with the same amount. The same procedure works on vertices. Suppose that we have a positive penalty $q(v, d_v)$ for all $d_v \in D_v$. Then by (3.2) we can decrease the penalty $q(v, d_v)$ with the minimum vertex penalty and add the same value to the objective. The condition that all penalties should be nonnegative is not really crucial, but allows us to maintain a lower bound on the objective value.

A special case are penalty matrices with the property that $p(v, d_v, w, d_w) = \bar{q}(v, d_v) + \bar{q}(w, d_w)$, i.e., the elements are the sum of values corresponding to the rows and columns. Then we can reduce all edge-penalties to zero, and thus remove the edge from the constraint graph by shifting all edge-penalties to the frequencies of the two corresponding vertices.

4.4.3 Domain reduction

In this section we devise methods to reduce the number of partial feasible solutions to the ones that are candidates to be used in optimal solutions. We describe two ways of doing so, namely upper bounding (in Section 4.4.3.1), and dominance (in Section 4.4.3.2).

4.4.3.1 Upper bounding

Upper bounding in its simplest form is performed on vertices as follows. Consider a vertex v and its neighbors $N(v)$. We want to derive an upper bound $u(v, \delta(v))$ on the total penalty incurred by node v in the optimal solution of the FAP, i.e., an upper bound on the vertex-penalty of v and the edge-penalties of the edges incident with v.

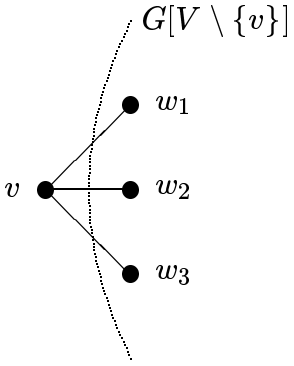
Consider an arbitrary partial solution $d^*_{N(v)} \in D_{N(v)}$. Then we compute the frequency for v with the lowest penalty:

$$P(d^*_{N(v)}) = \min_{d_v \in D_v} \left\{ q(v, d_v) + \sum_{w \in N(v)} p(v, d_v, w, d^*_w) \right\}$$

Among all possible choices for $d^*_{N(v)} \in D_{N(v)}$ we take the one with highest penalty, i.e.

$$u(v, \delta(v)) = \max_{d^*_{N(v)} \in D_{N(v)}} P(d^*_{N(v)})$$

Then the value $u(v, \delta(v))$ is certainly an upper bound on the penalty incurred by an optimal choice of frequency for v.



D_v	$q(v, d_v)$
1	0
2	0
3	0

$p(v, d_v, w, d_{w_1})$

	D_{w_1}	
D_v	1	2
1	2	0
2	0	2
3	1	1

$p(v, d_v, w, d_{w_2})$

	D_{w_2}	
D_v	1	2
1	0	2
2	2	0
3	1	1

$p(v, d_v, w, d_{w_3})$

	D_{w_3}	
D_v	1	2
1	1	0
2	0	1
3	0	0

Figure 4.7: Example upper bounding and dominance

We illustrate this upper bounding technique with the following example, see Figure 4.7. Let v be a vertex in the constraint graph G. Its domain contains three frequencies: 1, 2, and 3. It is connected to three vertices w_1, w_2, and w_3 each of which has two frequencies: 1 and 2. For all $d_v \in D_v$, the total penalty is $q(v, d_v) + \sum_{w \in \delta(v)} p(v, d_v, w, d^*_w)$ where d^*_w is the frequency chosen for w. In Table 4.1 we have computed for any combination of $(d_{w_1}, d_{w_2}, d_{w_3})$ the best frequency d^*_v for v, i.e., the one such that the total penalty

d_{w_1}	d_{w_2}	d_{w_3}	$q(v,d_v)+\sum_{w\in N(v)} p(v,d_v,w,d_w)$			
			$d_v=1$	$d_v=2$	$d_v=3$	best
1	1	1	5	0	2	0
1	1	2	4	1	2	1
1	2	1	3	2	2	2
1	2	2	2	3	2	2
2	1	1	3	2	2	2
2	1	2	2	3	2	2
2	2	1	1	4	2	1
2	2	2	0	5	2	0

Table 4.1: Penalties example upper bounding and dominance

is minimal among all possible frequencies for v. Table 4.1 shows that for this example the upper bound is 2, the maximum of the last column. In general, though not in this example, any frequency $d_v \in D_v$ with $q(v,d_v) > u(v,\delta(v))$ can be removed from D_v.

For arbitrary $v \in V$, we can compute this upper bound by solving an integer linear program. For all $w \in N(v)$, $d_w \in D_w$ we introduce a binary variable

$$y(w,d_w) = \begin{cases} 1 & \text{if } d_w \in D_w \text{ is assigned to } w \in N(v) \\ 0 & \text{otherwise} \end{cases}$$

If the variable z denotes the actual upper bound, the integer linear program reads as

$$u(v,\delta(v)) = \max \; z \tag{4.1}$$

$$\text{s.t.} \quad z \le q(v,d_v) + \sum_{\substack{w\in N(v)\\ d_w\in D_w}} p(v,d_v,w,d_w)y(w,d_w) \quad \forall d_v \in D_v \tag{4.2}$$

$$\sum_{d_w\in D_w} y(w,d_w) = 1 \qquad \forall w \in N(v) \tag{4.3}$$

$$y(w,d_w) \in \{0,1\} \qquad \forall w \in N(v), d_w \in D_w \tag{4.4}$$

The constraints (4.3) and (4.4) enforce that for each neighbor of v exactly one frequency is chosen. For a given choice of frequencies $d^*_{N(v)}$ the right-hand sides of constraints (4.2) are the penalties incurred with each of the corresponding frequencies for v. Thus, a frequency d_v with smallest penalty determines the highest value z can obtain for the particular choice of frequencies for the neighbors of v. For each possible assignment of frequencies to the neighbors of v we determine this value. The worst choice of $d^*_{N(v)}$ is one for which this value is maximal. This choice determines the value of z, and so $u(v,\delta(v))$.

Frequencies $d_v \in D_v$ for which $q(v, d_v) > u(v, \delta(v))$ can be removed from the domain. In the preprocessing phase such frequencies are removed for all vertices. Also, partial feasible solutions $d_S \in D_S$, $v \in S$, for which the penalty incurred by the frequency assigned to v and its incident edges is higher than $u(v, \delta(v))$ need not be considered.

The above technique can be generalized to sets of vertices, instead of single vertices. Consider a set $S \subset V$ with its set of assignments D_S.

$$u(S, \delta(S)) = \max \; z \tag{4.5}$$

$$\text{s.t.} \quad z \leq q(S, d_S) + \sum_{w \in N(S)} \sum_{d_w \in D_w} \sum_{v \in N(w) \cap S} p(v, d_v, w, d_w) y(w, d_w)$$

$$\forall d_S \in D_S \tag{4.6}$$

$$\sum_{d_w \in D_w} y(w, d_w) = 1 \qquad \forall w \in N(S) \tag{4.7}$$

$$y(w, d_w) \in \{0, 1\} \qquad \forall w \in N(S), d_w \in D_w \tag{4.8}$$

Here, $q(S, d_S)$ denote the total penalty involved by an assignment d_S, i.e.,

$$q(S, d_S) = \textstyle\sum_{v \in S} q(v, d_v) + \sum_{\{v,w\} \in E[S]} p(v, d_v, w, d_w)$$

This value is a lower bound on the total penalty involved in any complete assignment based on the partial assignment d_S. So, if $q(S, d_S) > u(S, \delta(S))$, then this partial assignment cannot be extended to an optimal complete assignment. Hence, it can be removed from the set of assignments D_S. An even better lower bound on the penalty in any complete assignment is given by the total penalty incurred by the subgraph $G[S]$ and the edges $\delta(S)$, i.e.

$$l(S, \delta(S), d_S) = q(S, d_S) + \textstyle\sum_{w \in N(S)} \min_{d_w \in D_w} \left\{ \sum_{v \in N(w) \cap S} p(v, d_v, w, d_w) \right\}$$

Whenever $l(S, \delta(S), d_S) > u(S, \delta(S))$, we can remove d_S from our set of assignments for S. We will call an assignment non-redundant if $l(S, \delta(S), d_S) \leq u(S, \delta(S))$.

The upper bound $u(S, \delta(S))$ is especially powerful if the number of edges in the cut-set $\delta(S,)$ is small, or if the sum of the maximum penalties incurred by the cut-set edges is not too large. If the upper bound $u(S, \delta(S)) = 0$ for a subset S, then we know that given any assignment to the vertices $V \setminus S$, the partial solution can be extended to a complete solution without additional penalty. This implies that we can remove the subset S and the edges $\delta(S)$ from the constraint graph.

A similar upper bounding technique can be applied to a small extension of the set S and the edges in its cut-set, i.e., an upper bound for the induced subgraph $G[S]$, the edges $\delta(S)$ and the vertices $N(S)$.

Note that, if $T \subseteq S$, $u(T, \delta(T)) \leq u(S, \delta(S))$, which implies that the upper bound for S is also valid for T. The upper bound $u(S, \delta(S))$ can also be used in combination with lower bounds. Let $S, T \subset V$ be disjoint subsets, and let $l(S)$ be a lower bound on the penalty incurred by $G[S]$. Then, an upper bound $u(T, \delta(T))$ is given by $u(T, \delta(T)) = u(T, \delta(T)) - l(S)$. Similarly, if $l(S, \delta(S))$ is a lower bound on the penalty incurred by $G[S]$ and the edges $\delta(S)$, then an upper bound for $G[T]$ is given by $u(T) = u(S \cup T, \delta(S \cup T)) - l(S, \delta(S))$.

The main problem with $u(S, \delta(S))$ is that it may take quite some time to compute it. It may be preferable to compute the value of some relaxation of (4.5)-(4.8). The LP-relaxation does not generate really powerful upper bounds. Our choice is therefore to relax (4.5)-(4.8) by taking a subset of the constraints (4.6), i.e., a number of partial feasible solutions with low $q(S, d_S)$. In case we restrict ourselves to one good partial solution d_S^* for S we can solve the relaxed problem by inspection, and use this as an upper estimate of $u(S, \delta(S))$:

$$\begin{aligned} u(S, \delta(S)) &\leq q(S, d_S^*) + \sum_{w \in N(S)} \sum_{d_w \in D_w} \sum_{v \in N(w) \cap S} p(v, d_v^*, w, d_w) y(w, d_w) \\ &= q(S, d_S^*) + \sum_{w \in N(S)} \max_{d_w \in D_w} \sum_{v \in N(w) \cap S} p(v, d_v^*, w, d_w) \end{aligned} \tag{4.9}$$

Note that good partial solutions are usually available through heuristics, or are generated in the dynamic programming algorithm.

4.4.3.2 Dominance

Upper bounding techniques are a quick way to eliminate the worst partial feasible solutions, but these techniques sometimes only remove a small fraction of the solutions that are redundant. In this subsection we develop techniques that remove partial solutions for which there exist better alternatives. Consider again the example of Figure 4.7. Though frequency 3 could not be removed from D_v using the upper bound, we can verify in Table 4.1 that for no choice of frequencies for the neighbors of v frequency 3 is the unique optimal choice. In other words, in any solution where frequency 3 is chosen we can replace it by another frequency without obtaining a worse solution. Therefore, we maintain at least one of the optimal solutions by removing this frequency from D_v.

The abstract concept of dominance is as follows. Let $v \in V$. Consider all partial solutions of $N(v)$. If these solutions can be extended with a frequency of $D_v \setminus \{d_v^*\}$ to solutions

at minimum cost, then d_v^* is not necessary to obtain an optimal solution. Therefore d_v^* can be removed from D_v. We say that d_v^* is *dominated* by the frequencies in $D_v \setminus \{d_v^*\}$. This concept can also be generalized to sets of vertices, similar to the generalization of the upper bounds to sets $S \subset V$. Let d_S^* be an assignment to S, then d_S^* is dominated by the other non-redundant assignments $D_S \setminus \{d_S^*\}$ if every partial feasible solution of $N(S)$ can be extended to a solution at minimum cost with an assignment of $D_S \setminus \{d_S^*\}$.

To find out whether d_S^* is dominated by $D_S \setminus \{d_S^*\}$, we formulate the following feasibility problem, which has a feasible solution if and only if d_S^* is the unique minimum for some choice of frequencies of the neighbors. Therefore, it is dominated if and only if this problem has no solution. The binary variables $y(w, d_w)$ used in this formulation have the same meaning as in the previous subsection: $y(w, d_w) = 1$ if frequency d_w is chosen for node w, and 0 otherwise. Then the feasibility problem reads

$$q(S, d_S^*) + \sum_{\substack{w \in N(S) \\ d_w \in D_w}} \left\{ \sum_{v \in N(w) \cap S} p(v, d_v^*, w, d_w) \right\} y(w, d_w)$$

$$< q(S, d_S) + \sum_{\substack{w \in N(v) \\ d_w \in D_w}} \left\{ \sum_{v \in N(w) \cap S} p(v, d_v, w, d_w) \right\} y(w, d_w) \; \forall d_S \in D_S \setminus \{d_S^*\} \tag{4.10}$$

$$\sum_{d_w \in D_w} y(w, d_w) = 1 \qquad \forall w \in N(S) \tag{4.11}$$

$$y(w, d_w) \in \{0, 1\} \qquad \forall w \in N(S) \forall d_w \in D_w \tag{4.12}$$

For any solution of $N(S)$ the constraints (4.10) state that the penalty of d_S^*, the left hand side (LHS), should be smaller than the penalty of each $d_S \in D_S \setminus \{d_S^*\}$, the right hand side (RHS). In other words, if there is a solution of $N(S)$ with this property, then d_S^* is the unique optimum for this solution, and thus it is *not* dominated by the other frequencies in $D_S \setminus \{d_S^*\}$.

To transform (4.10)-(4.12) into an integer linear program we introduce a variable z, which denotes the maximum difference between the RHS and LHS of (4.10), i.e., it is a measure of the "minimality" of d_S^*.

$$\max \quad z \tag{4.13}$$

$$\text{s.t.} \quad z \leq q(S, d_S) - q(S, d_S^*) + \sum_{w \in N(v)} \sum_{d_w \in D_w} \{ \sum_{v \in N(w) \cap S} \Delta p(v, d_v, d_v^*, w, d_w) \} y(w, d_w)$$

$$\forall d_S \in D_S \setminus \{d_S^*\} \tag{4.14}$$

$$\sum_{d_w \in D_w} y(w, d_w) = 1 \qquad \forall w \in N(S) \tag{4.15}$$

$$y(w, d_w) \in \{0, 1\} \tag{4.16}$$

where $\Delta p(v, d_v, d_v^*, w, d_w) = p(v, d_v, w, d_w) - p(v, d_v^*, w, d_w)$. Clearly, if $z > 0$, then d_S^* is not dominated, since the feasibility problem has a solution; otherwise, if $z \leq 0$, d_S^* is dominated.

The formulation (4.13)-(4.16) resembles the upper bound formulation (4.5)-(4.8). Moreover, this problem has to be solved for all non-redundant assignments. Therefore, we again relax the problem by removing constraints. For a good partial solution d_S we generate the corresponding constraint (4.14). This restricted problem can then be approximated by inspection. From (4.14) we get:

$$\begin{aligned} z &\leq q(S, d_S) - q(S, d_S^*) + \sum_{w \in N(v)} \sum_{d_w \in D_w} \{ \sum_{v \in N(w) \cap S} \Delta p(v, d_v, d_v^*, w, d_w) \} y(w, d_w) \\ &= q(S, d_S) - q(S, d_S^*) + \sum_{w \in N(v)} \max_{d_w \in D_w} \{ \sum_{v \in N(w) \cap S} \Delta p(v, d_v, d_v^*, w, d_w) \} \\ &\leq q(S, d_S) - q(S, d_S^*) + \sum_{\{v,w\} \in \delta(S)} \max_{d_w \in D_w} \Delta p(v, d_v, d_v^*, w, d_w) \end{aligned} \tag{4.17}$$

So, if the RHS of (4.17) is already ≤ 0, then d_S^* is dominated.

4.5 Iterative Version Algorithm

Both time and memory are insufficient to solve large instances with the dynamic programming algorithm described in Section 4.3, even if we use the reduction techniques of Section 4.4. During the algorithm, the number of non-redundant assignments explodes for these instances. We can point out two reasons. On the one hand, the width of our tree decomposition is too large. On the other hand, the number of frequencies available for a vertex is too large. In this section we focus on this last reason. Instead of assigning frequencies to the vertices, we propose to assign subsets of frequencies. So, we partition the domain of a vertex in a number of subsets, and assign one of them to the vertex. To handle these subsets as frequencies of a new MI-FAP, we have to harmonize the vertex and edge-penalties for all frequencies in a subset. We take as penalty the minimum of the individual penalties. In this way the solution value of the new MI-FAP is a lower bound for the original problem. We can extend this idea to an iterative method which provides a sequence of lower bounds for the original instance. The dynamic programming algorithm is used as a subroutine to solve the MI-FAPs with the substantially smaller domains. Contrary to the original MI-FAP, time and memory are sufficient to solve these MI-FAPs, because they are much smaller.

The idea of the method is that we identify a subset of the domain with each vertex. The vertex- and edge-penalties for these subsets are estimated from below. For example, consider the matrix of edge-penalties given in Figure 4.8(a). The level of interference on this edge is 10 if the difference between the frequencies is less than 2. If we divide the frequencies in two groups $\{1, 2\}$, and $\{3, 4\}$, we obtain 4 blocks in the table of edge-penalties with (almost) the same values. In most cases there is no difference between the penalties as long as the pairs of frequencies are in the same block. Therefore, let us construct a new MI-FAP in which we have to assign either the subset $\{1, 2\}$ or the subset $\{3, 4\}$ to the vertices. The edge-penalties in this new MI-FAP are given by the minimum of the values in each block (see Figure 4.8(b)). Solving this substantially smaller problem provides a lower bound for the optimal value of the original problem. The quality of the lower bound depends on the size of the blocks: many small blocks will provide a better lower bound than a small number of large blocks. In most real-life instances the block structure of the penalty matrices arises naturally, since the available frequencies for an antenna can be divided in groups of frequencies that are in the same part of the spectrum.

d_v, d_w	1	2	3	4
1	10	10	0	0
2	10	10	10	0
3	0	10	10	10
4	0	0	10	10

(a) original penalty matrix

d_v, d_w	$\{1, 2\}$	$\{3, 4\}$
$\{1, 2\}$	10	0
$\{3, 4\}$	0	10

(b) new penalty matrix

Figure 4.8: Example to illustrate the idea behind the iterative algorithm

This idea can be formalized in the following algorithm. We start with the original problem $P = (G, D, p, q)$ and we partition for every vertex $v \in V$ the domain D_v in an initial number of n_v subsets $D_v^1, \ldots, D_v^{n_v}$. This partition is, for example, based on a natural partition of the frequencies in groups of frequencies that are in the same part of the spectrum.

Next, we construct a new MI-FAP $P' = (G, D', p', q')$, with

- domains $D'_v = \{1, \ldots, n_v\}$ for all vertices $v \in V$,
- vertex-penalties $q'(v, i) = \min_{d_v \in D_v^i} q(v, d_v)$ for every vertex $v \in V$, $i \in D'_v$, and
- edge-penalties $p'(v, i, w, j) = \min_{d_v \in D_v^i} \min_{d_w \in D_w^j} p(v, d_v, w, d_w)$ for all $\{v, w\} \in E$, $i \in D'_v$, $j \in D'_w$.

So, P' is defined on the same graph as P, and the domains of P' correspond with the subsets D_v^i, $i = 1, \ldots, n_v$. Since the vertex and edge-penalties in P' are the minimum of the penalties in the corresponding subset(s), the optimal value of the problem P' provides a lower bound for the optimal value of the original problem P.

Our way to obtain a sequence of non-decreasing lower bounds is based on an iterative refinement of the domain-subsets. A partition $\tilde{D}_v^1, \ldots, \tilde{D}_v^m$ of a domain D_v is called a refinement of another partition $\bar{D}_v^1, \ldots, \bar{D}_v^n$, if for every subset $\tilde{D}_v^i$, $i = 1, \ldots, m$, there exists a subset $\bar{D}_v^j$, $j \in \{1, \ldots, n\}$ in the second partition for which $D_v^i \subseteq D_v^j$. If $\tilde{P}$ and $\bar{P}$ are MI-FAPs corresponding to these partitions, then the value of the optimal solution of $\tilde{P}$ will be at least as high as the value of the optimal solution of $\bar{P}$, which implies that $\tilde{P}$ provides a lower bound that is greater than or equal to the lower bound provided by $\bar{P}$.

Now, we can extend the algorithm to obtain a lower bound to an algorithm that provides a sequence of non-decreasing lower bounds as follows. We construct a problem P' which provides us with the first lower bound. Next, we refine the partition of the subsets, and again construct a MI-FAP P' which hopefully provides us with a better lower bound. We can repeat the refinement of the partition as long as the efforts to solve the problem P' is reasonable in both time and memory. A flowchart of this algorithm is presented in Figure 4.9.

Whatever refinement procedure (i.e., for which vertices do we refine the partition, and how do we refine the partition) we apply, a guarantee that the new lower bound will be strictly greater than the old lower bound cannot be given in general. However, if for all vertices $v \in V$, the domain-subset that corresponds to the optimal solution of P' is not partitioned in the refinement procedure, then the 'old' optimal solution is still optimal in the new problem P'. This implies that a refinement can only be effective if at least one selected domain-subset is refined. Therefore, for each refinement we select one vertex v, for which we partition the assigned subset. To speed up the process in practice, we do not apply the dynamic programming algorithm after each single refinement, but after the refinement of the domains for a subset of the vertices $S \subseteq V$.

For a partition of the assigned subset for a vertex $v \in V$ we can compute an upper bound on the increase of the value of P'. This upper bound is used as criteria to select a partition. Consider the example of Figure 4.10. Let $D'_v = \{d_v^1, d_v^2\}$ be the assigned subset to v, and let D'_u and D'_w be the assigned subsets to the neighbors u and w, respectively. The total penalty incurred by this assignment is 0. However, if we either assign d_v^1 or d_v^2 to v, then the total penalty will be one. Hence, partition of the subset may lead to an increase of the value of P'. It cannot be guaranteed however, since the new optimal assigned may select a subset other than $\{d_v^1\}$ or $\{d_v^2\}$.

In general, an upper bound on the increase of the optimal value by a partition of the assigned subset can be computed as follows. We restrict ourselves to a partition of the

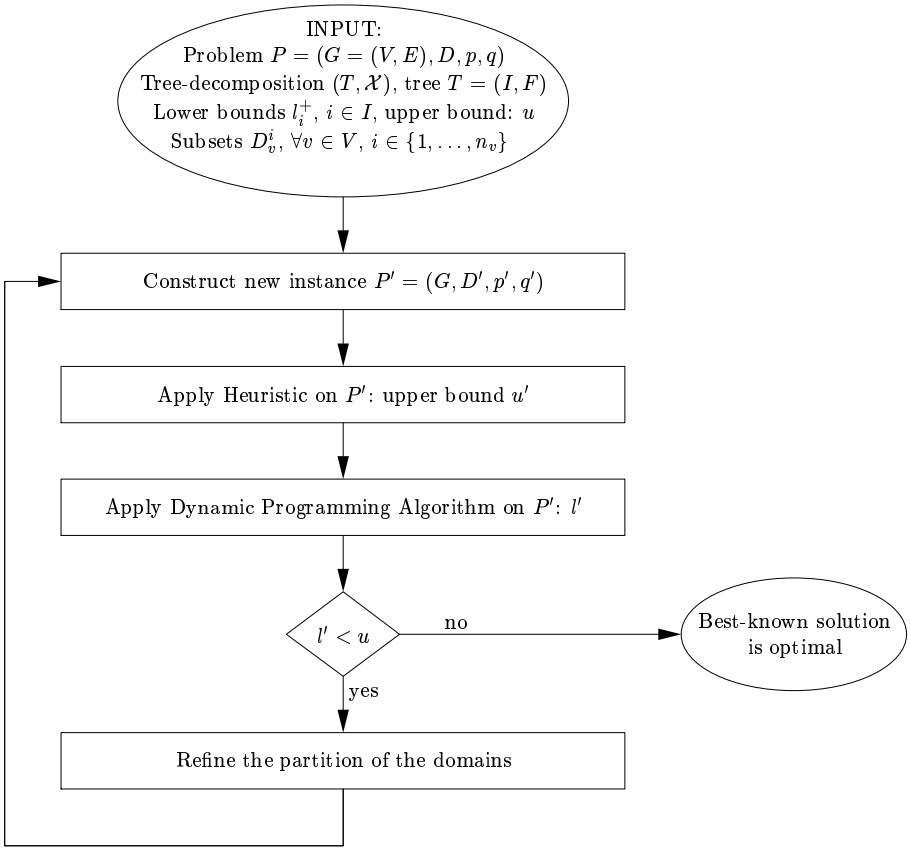
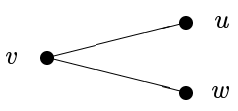


Figure 4.9: Iterative version of the algorithm

assigned domain-subset in two domain-subsets, but the procedure can easily be extended to a partition in more than two domain-subsets. The procedure can also be generalized to subsets of vertices instead of single vertices. Let $v \in V$, and D'_v be the domain-subset that corresponds to the optimal assignment. If we partition D'_v in A_v and $D'_v \setminus A_v$, then the value of the problem P' will increase with at most $\Delta\pi(v, A_v)$,

$$\Delta\pi(v, A_v) = \min\{\pi(v, A_v), \pi(v, D'_v \setminus A_v)\} - \pi(v, D'_v)$$



penalties		D'_u	D'_w	total
D'_v	d_v^1	1	0	1
	d_v^2	0	1	1
minimum		0	0	1

Figure 4.10: Example to illustrate the partition of assigned subsets

where

$$\pi(v, D) = \min_{d_v \in D} q(v, d_v) + \sum_{w \in N(v)} \min_{d_v \in D} \min_{d_w \in D'_w} p(v, d_v, w, d_w)$$

Among all partitions A_v, $D'_v \setminus A_v$, the best partition, according to the value $\Delta\pi(v, A_v)$, is $A^*_v = \arg\max_{A_v \subset D'_v} \Delta\pi(v, A_v)$. If $\Delta\pi(v, A^*_v) = 0$ then no single refinement of the partition for vertex v will result in an increase of the lower bound for P. Therefore, the subset S for which we will partition the assigned subset is given by the vertices for which $\Delta\pi(v, A^*_v) > 0$.

The iterative method can be separated from the dynamic programming algorithm. In principle we can use any exact algorithm to solve the consecutive MI-FAPs. However, the use of the dynamic programming algorithm of Section 4.3 enables us to use information of previous solved problems. More precisely, during the computation of the optimal solution of a previous problem P', we obtain for all $i \in I$ a lower bound $l(Y_i, \delta(Y_i))$ for the penalty incurred by $G[Y_i]$ and the edges $\delta(Y_i)$. These values are also lower bounds on the penalty in the new problem P', which implies that we can compute upper bounds $u(V \setminus Y_i) = u' - l(Y_i, \delta(Y_i))$ for all $i \in I$. Here, u' is a general upper bound for the new problem P' which can be computed by one of the heuristics available for the MI-FAP. If the increase of u' is not too large for two consecutive problems P', then the upper bounds for the subsets are often relatively strong.

4.6 Computational Results

In this section we report on the results we have obtained using the approach described in the previous sections. We tested the methods described in this chapter on real-life instances obtained from the CALMA-project (cf. the Sections 2.2.4 and 2.6). The set of instances consists of two parts. The CELAR instances are real-life problems from a military application. The GRAPH instances are randomly generated problems with the same characteristics. We only used the 11 so-called penalty-instances, since for the other instances the objective is either to minimize the frequency span, or the minimize the number of frequencies used. In this section we solve 7 out of the 11 instances to optimality and we obtain good lower bounds for the other instances.

The solution procedure can be divided in four parts, each of which is analyzed in the forthcoming subsections. In Section 4.6.1 we report on the results obtained with the preprocessing techniques of Section 4.4. The results of the heuristic to construct a tree decomposition of Section 4.2 are presented in Section 4.6.2. In Section 4.6.3, we show that some of the instances of the CALMA-project can be solved to optimality with the dynamic programming algorithm of Section 4.3. Furthermore, we compare the performance of the

dynamic programming algorithm with the polyhedral approach on 5 small test instances that have been constructed from one of the CELAR instances. Section 4.6.4 is devoted to the lower bounds which were obtained with the iterative version of the algorithm described in Section 4.5. Finally, in Section 4.6.5 we combine the iterative algorithm of Section 4.5 with the integer programming techniques of Chapter 3 in order to improve the lower bounds even further.

All implementations have been carried out in C++. The programs for the dynamic programming algorithm and the iterative version of the algorithm were running on a DEC 2100 A500MP workstation with 128Mb internal memory. The programs for preprocessing, for the construction of a tree decomposition, and for the computation of upper bounds for single vertices were executed on a Pentium II - 233 Mhz Personal Computer with 32Mb internal memory. We used the callable library of CPLEX 4.0 to solve (integer) linear programming problems.

4.6.1 Preprocessing

We start our computations with the application of the graph and domain reduction techniques described in Section 4.4. The following procedure is repeated as long as the size of the problem is reduced. First of all, we apply penalty shifting from edges to vertices and from vertices to the objective. Next, we apply the graph reduction techniques: the removal of vertices with only one domain element, the removal of edges with only zero penalty, and the removal of vertices of degree less than or equal to two. Then, we calculate the upper bound (4.9) for every vertex, and we apply the dominance test (4.17) for single vertices. If a frequency is dominated, then we remove this frequency from the domain. The dominance test (4.13)-(4.16) with $S = \{v\}$ yields no additional reduction. If, due to the upper bound and dominance test, a vertex with only one frequency occurs, we remove the vertex. We apply the same upper bound (4.9) and dominance test (4.17) for adjacent vertices. Contrary to the dominance test for a single vertex we cannot remove the frequencies of the combination in case it is dominated. Therefore, we increase the edge-penalty of this combination with an amount that guarantees that it will never occur in a non-redundant assignment. Moreover, if given a frequency $d_v \in D_v$, the combination (d_v, d_w) is dominated for all $d_w \in D_w$, we can remove the frequency d_v from the domain D_v.

In the Table 4.2 statistics for all penalty-instances before and after preprocessing are reported. Successively, we report the number of vertices ($|V|$) and the number of edges ($|E|$) in the constraint graph, and the average number of domain elements ($|D|$). In addition, we report the value that is fixed by the preprocessing phase, the best known value (see Kolen [118]), and the best known lower bound (cf. Aardal et al. [1] and Hurkens and Tiourine [184]). For the GRAPH instances this lower bound is not available. Table 4.2

	before preprocessing			after preprocessing				best	previous
instance	$\|V\|$	$\|E\|$	$\|D\|$	$\|V\|$	$\|E\|$	$\|D\|$	fixed	value	lower bound
CELAR 06	100	350	39.9	82	327	39.9	0	3,389	5
CELAR 07	200	817	39.9	162	764	34.6	0	343,592	5
CELAR 08	458	1,655	39.5	365	1,539	39.4	0	262	0
CELAR 09	340	1,130	39.5	67	165	35.6	11,391	15,571	14,969
CELAR 10	340	1,130	39.5	0	0	-	31,516	31,516	31,204
GRAPH 05	100	416	37.1	0	0	-	221	221	-
GRAPH 06	200	843	37.7	119	348	16.2	4,112	4,123	-
GRAPH 07	200	843	36.7	0	0	-	4,324	4,324	-
GRAPH 11	340	1,425	37.7	340	1,425	32.6	2,553	3,080	-
GRAPH 12	340	1,255	37.6	61	123	15.3	11,496	11,827	-
GRAPH 13	458	1,877	38.4	456	1,874	38.1	8,676	10,110	-

Table 4.2: Statistics and preprocessing CALMA-instances

shows that 3 out of the 11 penalty-instances are solved by preprocessing only. For the instances CELAR 10 and GRAPH 07 this is mainly due to the vertex penalties, that cause the removal of many frequencies. The graph reduction in the instances CELAR 09 and GRAPH 12 can be explained in the same way. Table 4.2 also shows that there is a major difference between the real-life CELAR instances and the randomly generated GRAPH instances. The fixed value for the CELAR instances without vertex-penalties is simply zero, whereas for the GRAPH instances 80% or more of the best known value can be fixed. This difference can be explained by the effectiveness of the different preprocessing rules. For the CELAR instances, the main part of the reduction is due to the removal of vertices with degree less than or equal to two, whereas the main part of the reduction for the GRAPH instances is due to penalty shifting (fixing) and the dominance test (4.17) for single vertices. In fact, for the instance GRAPH 05, a first round of shifting penalties resulted in a lower bound of 220. As a consequence, many domain elements could be removed from the problem, and the constraint graph reduced substantially. A new round of shifting penalties resulted in the proof of optimality of the best known solution. The running time of the preprocessing phase is within a minute for all penalty-instances.

4.6.2 Construction of Tree-decompositions

The second step in solving a MI-FAP is the construction of a tree decomposition of the preprocessed constraint graph. In Table 4.3 we report on the results of the heuristic of Section 4.2. We also report the maximum clique size minus one. Since every clique should

instance	$\|V\|$	$\|E\|$	width	max clique - 1	cpu-time (sec)
CELAR 06	82	327	11	10	17
CELAR 07	162	764	17	10	176
CELAR 08	365	1,539	18	10	802
CELAR 09	67	165	7	7	0
GRAPH 06	119	348	17	5	137
GRAPH 11	340	1,425	104	7	19,749
GRAPH 12	61	123	4	4	6
GRAPH 13	456	1,874	133	6	63,586

Table 4.3: Construction of a tree decomposition

be in at least one node of the tree, this value is a lower bound for the treewidth of a graph. Table 4.3 shows that the gap between the width and the lower bound varies from zero for small instances to very large for the large GRAPH instances. For these instances it is not clear which bound is poor. We have tried several variants of our heuristic to improve the width of the tree decomposition, but without any success. Based on the tree decomposition we were able to represent the constraint graph of CELAR 06 in Figure 4.11. The figure shows that the graph can be decomposed by several small separating vertex sets, which validates the ideas behind the heuristic of Section 4.2.

4.6.3 Dynamic Programming Algorithm

In this subsection we report the computational results obtained with the dynamic programming algorithm of Section 4.3. The order in which we calculate all non-redundant assignments for the subsets Y_i, $i \in I$ is based on the available upper bounds (4.9) for $S = Y_i$. The sets Y_i are ordered according to non-decreasing $u(Y_i, \delta(Y_i))$. During the dynamic programming algorithm the upper bounds for the subsets are updated every time we obtain a new lower bound for a subset by computing all non-redundant assignments. If an upper bound for a subset decreases and all non-redundant assignments are already computed, we remove all assignments with penalty larger than the new upper bound. The dynamic programming algorithm is used with and without applying a dominance test for the subsets Y_i. As dominance test, we solve the linear programming relaxation of (4.13)-(4.16) with a limited number of constraints (4.14).

A first test of the dynamic programming algorithm is performed on 5 instances with size of the domains between 2 and 6 for all vertices. These instances were constructed from the instance CELAR 06 by taking a subset of the domain elements of predefined size. In Koster, van Hoesel and Kolen [121] the polyhedral approach is tested on these instances.

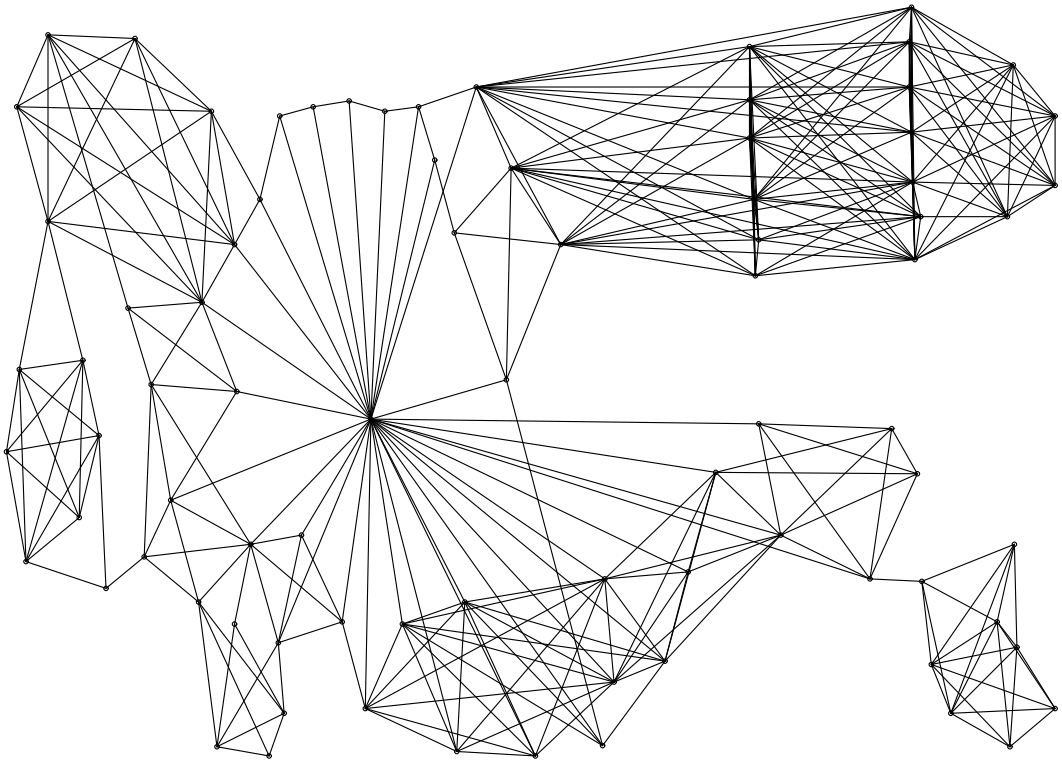


Figure 4.11: Graphical representation CELAR 06 based on computed tree decomposition

The tree decomposition approach is tested on these instances with and without using the dominance test (4.13)-(4.16). In Table 4.4 the computation times of the polyhedral method and the tree decomposition approach are compared. Without using dominance the dynamic programming algorithm cannot solve the largest instance. At some point during the dynamic programming algorithm the number of non-redundant assignments for a subset is too large to store into the memory of our computer. The table shows that both dynamic programming algorithms are competitive or substantially faster than the polyhedral method. We also may conclude that the use of the dominance test in the dynamic programming algorithm speeds up the process for these instances.

The dynamic programming algorithm is also performed on the original penalty-instances. The polyhedral method is not able to solve these instances, or even to generate non-trivial lower bounds. Table 4.5 shows the results that are obtained with the dynamic programming algorithm without dominance. Experiments with the dominance test did not result in a better performance of the algorithm for these instances. The instances CELAR 09, GRAPH 06 and GRAPH 12 can be solved very efficiently with this method. After more than 7.5 hours the algorithm was able to prove that the best known solution was optimal for this instance as well. Figure 4.12 shows the number of non-redundant assignments during the process compared with the theoretical number. The optimal value for all

instance	$\|D_v\|$	CPU-time for		
		polyhedral method	DP without dominance	DP with dominance
CELAR6a	2	3.5	3.8	8.8
CELAR6b	3	84.1	38.4	35.1
CELAR6c	4	11,785.5	408.6	219.2
CELAR6d	5	22,501.2	2,306.5	592.9
CELAR6e	6	75,570.5	-	2,399.4

Table 4.4: Computational results dynamic programming algorithm test instances

instance	optimal value	CPU-time (sec)
CELAR 06	3,389	27,102
CELAR 07	-	-
CELAR 08	-	-
CELAR 09	15,571	23
GRAPH 06	4,123	29
GRAPH 11	-	-
GRAPH 12	11,827	11
GRAPH 13	-	-

Table 4.5: Computational results dynamic programming algorithm

these instances is equal to the best known. The instance CELAR 06 is more difficult to solve. Mainly due to limitations in computer memory, we are not able to solve the other instances.

4.6.4 Iterative version

Table 4.5 in the previous subsection shows that the dynamic programming algorithm is not able to solve several instances. For these problems we apply the iterative version of the algorithm of Section 4.5. Before we start our computations we have to partition all domains in an initial number of subsets. In our experiments we start with either 2 or 4 subsets for every vertex. The partition of the subsets is based on a natural partition of the frequencies in the radio spectrum.

In each iteration of the algorithm, first a heuristic is applied to obtain an upper bound for the instance. In our computational experiments we used the genetic algorithm developed by Kolen [118]. Then, we apply the dynamic programming algorithm in the same way as

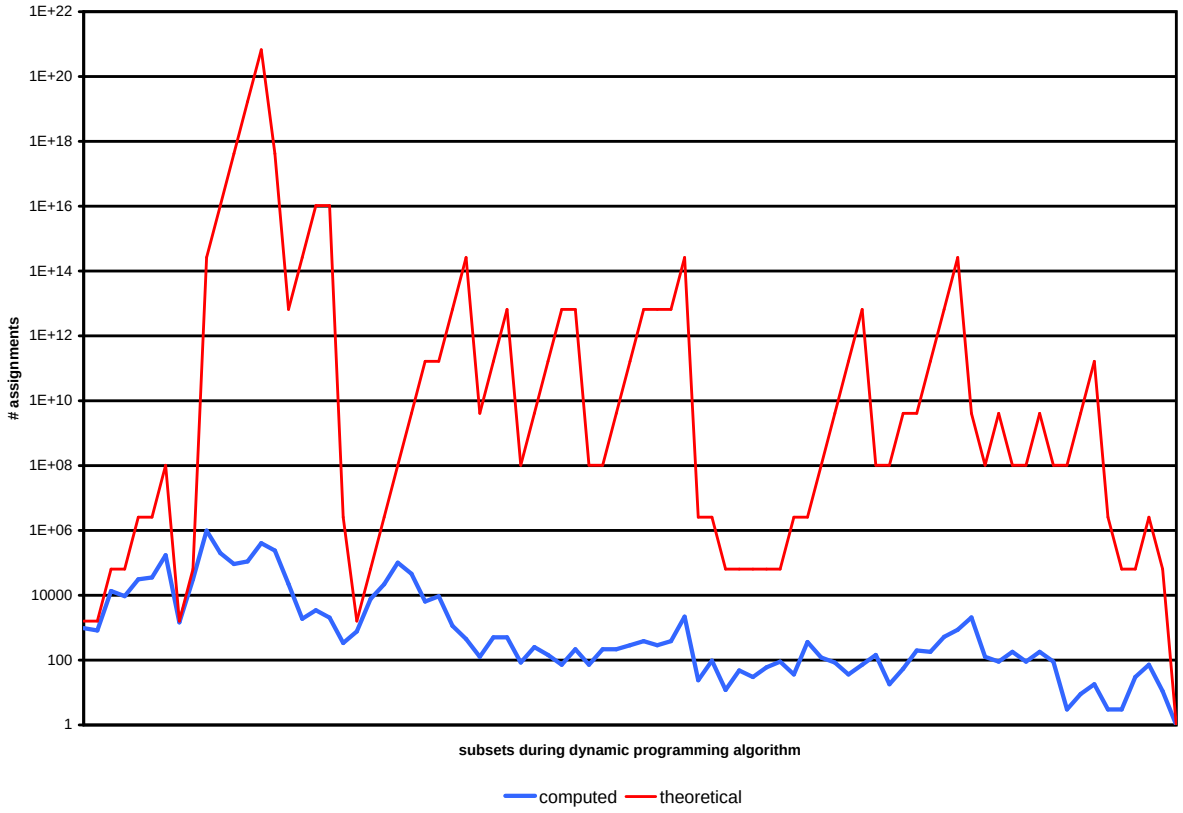


Figure 4.12: Number of non-redundant assignments `CELAR 06`

in the previous subsection. As last step in an iteration, we partition the selected domain-subset of the vertices in the set S. As described in Section 4.5, we base our selection of S and the actual partitions on the values $\Delta\pi(v, A_v^*)$. We limit the set S to at most 20 vertices. Moreover, we do not compute $\Delta\pi(v, A_v)$ for every partition of the selected domain-subset D'_v, but only for partitions of the form $\{1, \ldots, i\}, \{i+1, \ldots, |D'_v|\}$.

In Table 4.6 we report the results we obtained in this way for the instances that we could not solve with the original dynamic programming algorithm. For `CELAR 07` we obtain a lower bound that is within 12.5% of the best known value. Both for `CELAR 07` and `CELAR 08` the values are the first non-trivial lower bounds. For the instances `GRAPH 11` and `GRAPH 13`, the width of the tree decomposition is too large to apply the dynamic programming algorithm with any success. We also apply the iterative version to the instance `CELAR 06`. If we either start with an initial number of subsets of 2 or 4, we obtain a lower bound that is one away from optimal in third the time (or half the time) that is needed to solve the problem to optimality with the original dynamic programming algorithm. Figure 4.13 shows the improvement of the lower bound during the process. In case we start with 2 subsets per vertex we need 38 iterations to achieve the lower bound of 3388, whereas if we start with 4 subsets per vertex we need 35 iterations. Figure 4.14 shows a histogram

instance	initial # subsets	lower bound after preprocessing	lower bound iterative algorithm	upper bound	CPU-time (sec)
CELAR 06	2	0	3,388	3389	13,734
	4		3,388		9,429
CELAR 07	2	0	243,066	343592	259,022
	4		300,000		275,736
CELAR 08	2	0	87	262	313,168
	4		74		12,482
GRAPH 11	4	2,553	-	3,080	-
GRAPH 13	4	8,676	-	10,110	-

Table 4.6: Computational results iterative version of the algorithm

with the vertices as a function of the number of subsets after the last iteration. It shows that only for a restricted number of vertices the domain is refined during the process. In Figure 4.15 the maximum number of non-redundant assignments is displayed for each iteration. The figure shows that the algorithm that started with 4 domain subsets needs in the end substantially more non-redundant assignments to compute the lower bound than the algorithm that started with 2 domain elements.

4.6.5 Iterative Algorithm and Integer Programming

As mentioned in Section 4.5, the iterative algorithm can be separated from the dynamic programming algorithm. In this section we present the results of preliminary computational experiments to combine the iterative algorithm with the polyhedral approach of Chapter 3. From the computational results of Chapter 3 we know that the MI-FAP / PCSP can be solved through integer programming as long as the domains are small. The problem P' that has to be solved within an iteration of the iterative algorithm, contains many small domains, and therefore suits to be solved by integer programming. A straightforward implementation of this idea resulted in the lower bounds presented in Table 4.7. Table 4.7 shows that especially for the instances where tree decomposition fails (GRAPH 11 and GRAPH 13), the integer programming approach can improve the lower bounds. If we start with a partition in 4 subsets for all vertices, lower bounds of 98% of the best known value are obtained for both GRAPH 11 and GRAPH 13. Also for the instance CELAR 08, we can improve the lower bound substantially, from 33% to 57% of the best known solution. For the other instances the results are competitive with the iterative version of the dynamic programming algorithm.

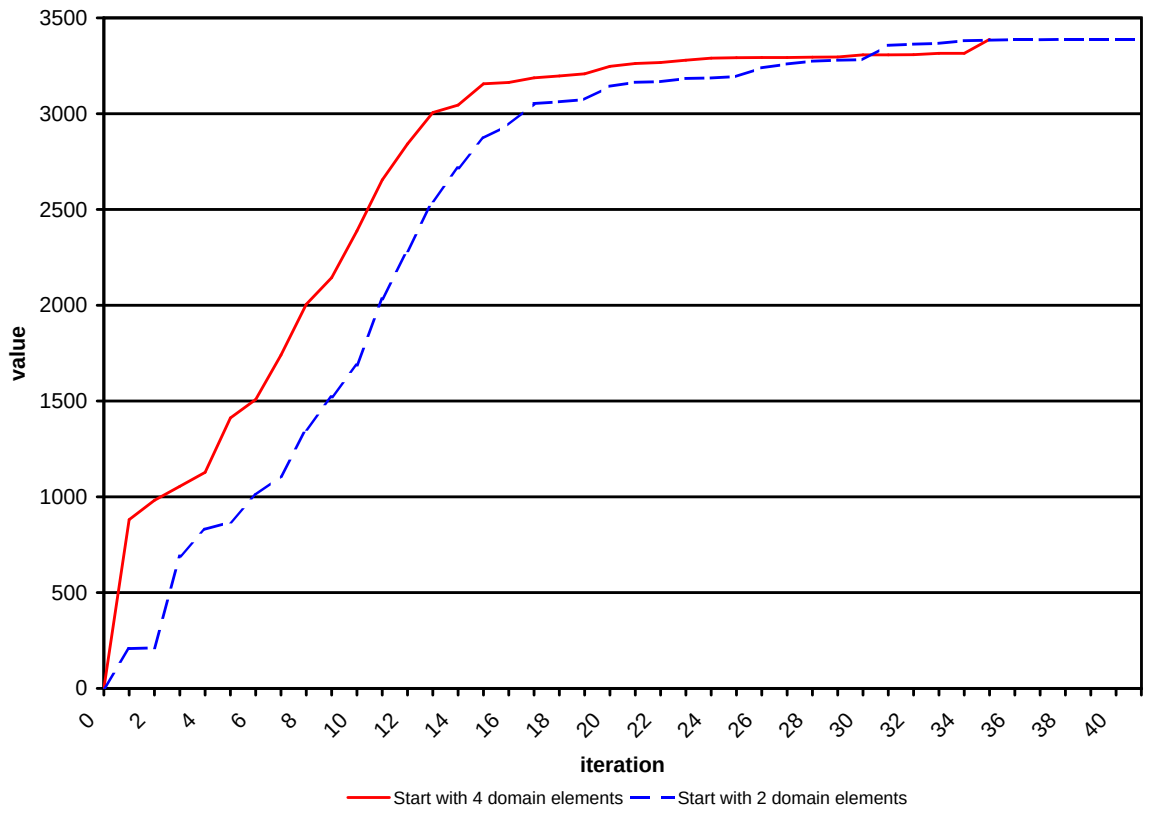


Figure 4.13: Lower bounds `CELAR 06`

4.7 Concluding Remarks

In this chapter we described a computational study to use the concept of tree decomposition to solve frequency assignment problems to optimality. We showed that the method, although theoretically polynomial in both time and space requirements, can only be applied to real-life problem instances if we use additional reduction techniques. The reduction techniques used include graph reduction, upper bounding and dominance of domain elements / partial assignments. Even with these techniques, it is not sure that the instances can be solved. Therefore, we presented an iterative version of the algorithm which can be used to obtain lower bounds for most of the instances. For a set of real-life instances, we proved optimality for several instances, whereas we obtained the first non-trivial lower bounds for the other instances. The iterative version of the algorithm can also be combined with the polyhedral approach of Chapter 3 resulting in even better lower bounds for the instances with large treewidth.

Based on these results, we state four directions for further research. One way is to embed either the dynamic programming algorithm or the iterative algorithm in a branch-and-bound framework. Hopefully, this results in even better lower bounds or even optimal

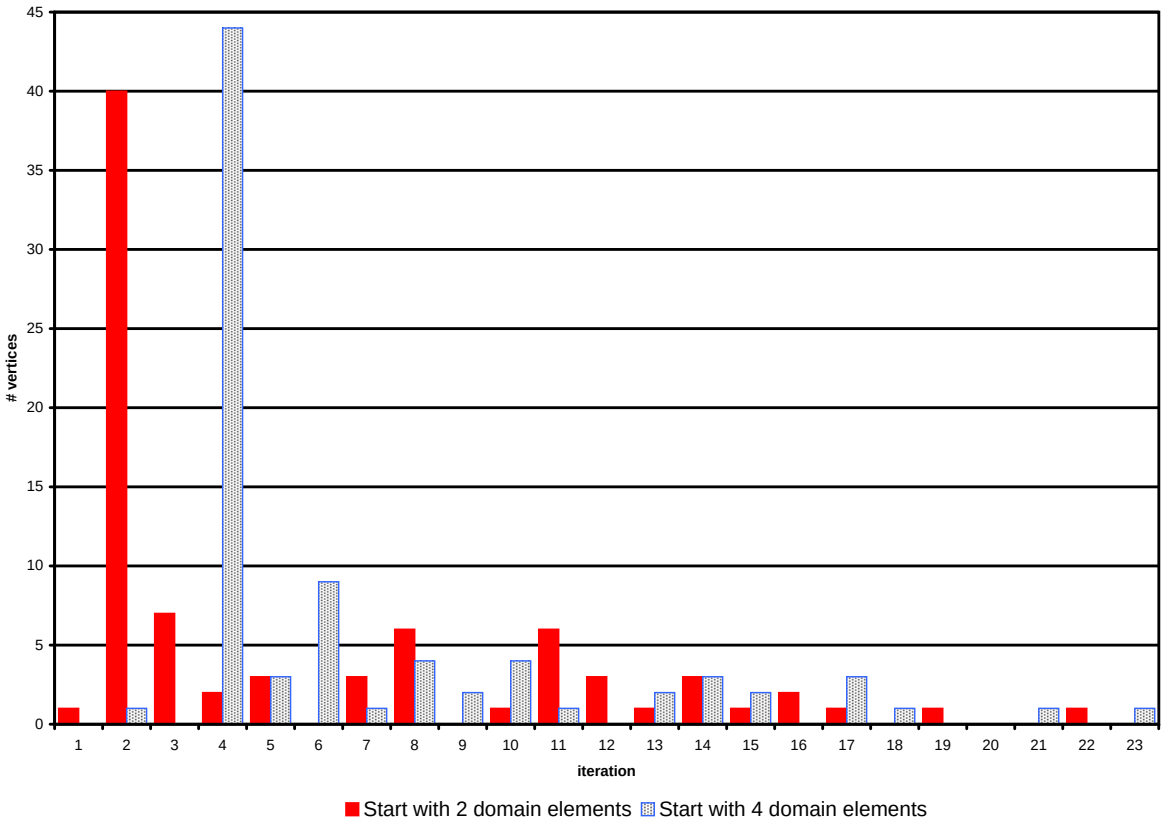


Figure 4.14: Number of vertices as function of number of domain-subsets for CELAR 06 at the end of the iterative algorithm.

solutions. Another way for further research is the application of this method to other hard combinatorial optimization problems. It is worthwhile to investigate the possibilities of this method for problems that are based on a graph, and which cannot be solved by the current solution methods.

A third direction for further research is the iterative algorithm. The preliminary computational results of Section 4.6.5 showed the potential power of this method for the MI-FAP. More sophisticated implementations of the algorithm may lead to even better results for the MI-FAP. Also application of the iterative method to other combinatorial optimization problems should be considered. Finally, research in the direction of (practical) algorithms to find a tree-decomposition with small treewidth should be carried out. The results of this chapter show that decreases in the width of tree decompositions will result in performance improvements of the dynamic programming algorithm. Both heuristics and exact methods should be considered to improve the tree decomposition of a graph.

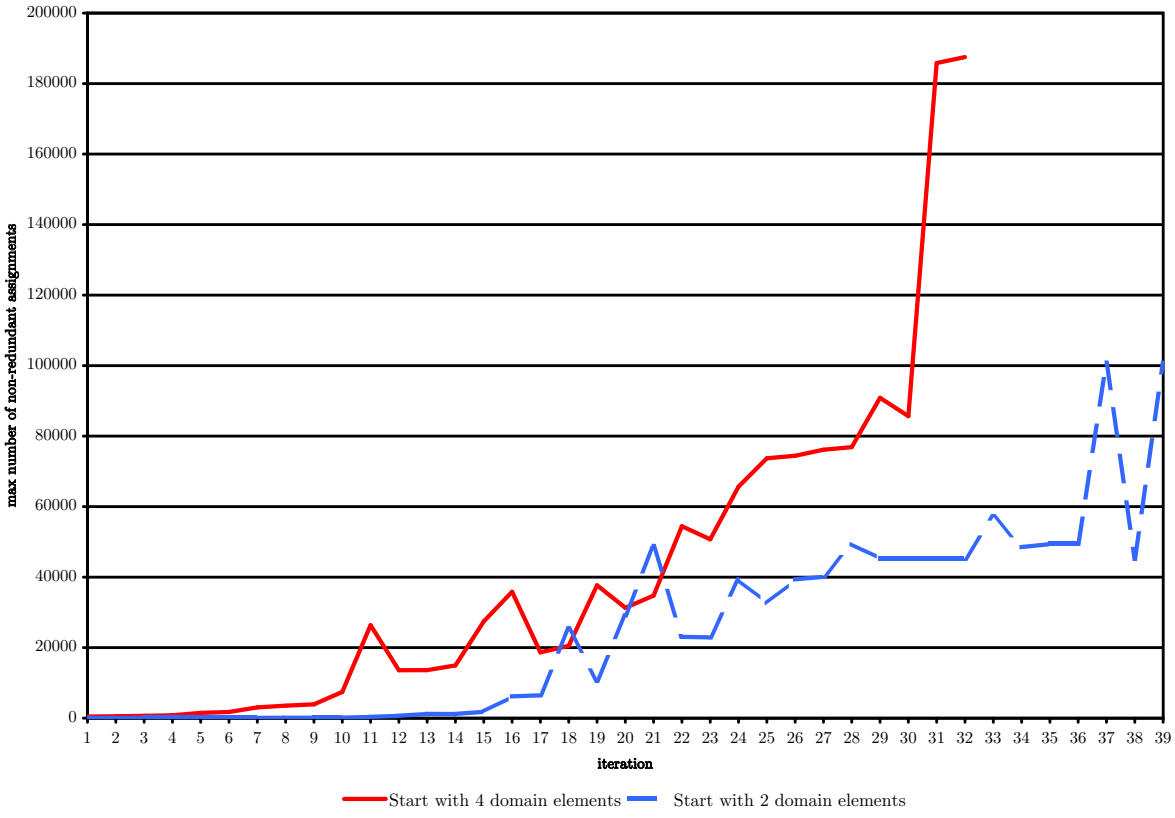


Figure 4.15: Maximum number of non-redundant assignments in every iteration for `CELAR 06`

instance	initial # subsets	lower bound tree decomposition	lower bound integer programming	upper bound	CPU-time (sec)
CELAR 06	2	3,388	2321	3389	69,470
	4	3,388	2146		67,904
CELAR 07	2	243,066	180,525	343592	256,418
	4	300,000	-		-
CELAR 08	2	87	125	262	346,318
	4	74	150		180,326
GRAPH 11	2	2553*	2898	3,080	70,864
	4		3016		74,113
GRAPH 13	2	8676*	9925	10,110	23,211
	4		9183		67,600

Table 4.7: Computational results iterative algorithm combined with the integer programming techniques of Chapter 3. Values indicated with a * are obtained by preprocessing (cf. Table 4.6).

5. Local Search Approaches

Up to now, we have concentrated on exact solution methods for the minimum interference frequency assignment problem (MI-FAP). The methods of the Chapters 3 and 4 provide (in theory) optimal solutions to the MI-FAP, or more general to the Partial Constraint Satisfaction Problem (PCSP). However, the computational results of both chapters show that the techniques are not strong enough to solve the more difficult (larger) real-life benchmark instances. In view of these results and the $\mathcal{NP}$-completeness of the problem, we cannot expect that the even larger real-world instances can be solved to optimality within the foreseeable future. Nevertheless, this does not mean that the techniques derived in the previous chapters can only be used to obtain lower bounds and to solve small instances. In this chapter we show that both methods, the polyhedral techniques, and the tree decomposition approach can be useful for the development of heuristics that generate solutions of high quality to the problem.

The heuristics we propose, are local search algorithms. In Section 5.1 we introduce the necessary notation. Next, in Section 5.2 we propose a local search algorithm that uses the integer programming results of Chapter 3. In Section 5.3, we show that also the results of the tree decomposition approach of Chapter 4 can be incorporated within a local search algorithm. For both algorithms preliminary computational results are presented. The chapter is closed with some concluding remarks in Section 5.4.

5.1 Preliminaries

Before we describe the actual local search approaches for the MI-FAP, we have to introduce some general notation that defines a local search algorithm. Any combinatorial optimization problem can be defined by a pair $(\mathcal{S}, f)$ where $\mathcal{S}$ is the set of solutions and f a cost function $f : \mathcal{S} \mapsto \mathbb{Z}$ that adds to each solution $s \in \mathcal{S}$ a cost $f(s)$. We assume that the objective is to find a solution with minimal cost. A local search framework is defined by its neighborhood function $\mathcal{N} : \mathcal{S} \mapsto 2^{\mathcal{S}}$. A solution $s \in \mathcal{S}$ is called locally optimal with respect to its neighborhood function $\mathcal{N}$ if $f(s) \leq f(s')$ for all $s' \in \mathcal{N}(s)$. A local search algorithm changes a given solution s to a locally optimal solution t. Starting with a solution s, either there exists an $s' \in \mathcal{N}(s)$ with $f(s') < f(s)$ or s is locally optimal. In case there exists an $s' \in \mathcal{N}(s)$ with $f(s') < f(s)$, then the solution s is replaced by s', and the search for a better solution in the new neighborhood $\mathcal{N}(s')$ is applied. In case no

s' exists, s is returned as the locally optimal solution.

In many implementations of local search, the framework is complemented with a disturbance function $\mathcal{D} : \mathcal{S} \mapsto 2^{\mathcal{S}}$. This function is used to escape from local minima: let $s \in \mathcal{S}$ be a locally optimal solution and the best solution found so far, then s is stored, and the local search algorithm is applied to an arbitrary $s' \in \mathcal{D}(s)$. In case the new local optimum s'' has objective value $f(s'') < f(s)$, then the solution s is removed, and s'' is stored as new best solution. The algorithm stops if no better local minimum is found in the last K iterations (disturbances and local optimizations), or the maximum computation time / maximum number of iterations is achieved. For an overview on local search we refer to Aarts and Lenstra [8].

For the PCSP the set of solutions $\mathcal{S}$ contains all possible assignments $(d_v)_{v \in V}$ of domain elements $d_v \in D_v$ to the vertices. So, the number of solutions, $|S|$, equals $\Pi_{v \in V} |D_v|$. The cost function f is just defined by the total sum of vertex and edge penalties. A simple neighborhood function for the PCSP maps an assignment $s = (d_v)_{v \in V}$ to all assignments $s' = (d'_v)_{v \in V}$ that differ from $(d_v)_{v \in V}$ for only one vertex, i.e. $s' \in \mathcal{N}(s)$ if and only if there is a $u \in V$ with $d'_u \neq d_u$, and for all $v \in V \setminus \{u\}$, $d'_v = d_v$. For reasons of conformity with local search approaches to other combinatorial optimization problems, we refer to this neighborhood function in the sequel as 1-Opt. In the next sections we describe more sophisticated neighborhood functions, and compare them with the neighborhood function 1-Opt.

Disturbance functions for the PCSP disturb either the assignment of a fixed number of the vertices, the assignment of a fixed percentage of the vertices, or the assignment of every vertex with a fixed probability. In our local search algorithms we use a disturbance function that changes the assignment of two arbitrarily selected vertices to a randomly generated domain element.

5.2 Local Search and Integer Programming

The computational results of Chapter 3 show that PCSPs with 2 elements per domain can be solved efficiently by integer programming techniques. Therefore, it is possible to incorporate this method within a heuristic to obtain good solutions. In fact, Kolen [118] first implemented this idea within a genetic algorithm. For the CALMA benchmark instances the best known solutions are obtained in this way (cf. Section 2.6). In this section we use the same idea within a local search framework. In Section 5.2.1 we propose two neighborhood functions based on this idea, whereas in Section 5.2.2 these functions are tested for the CALMA instances.

5.2.1 Neighborhood

In the local search algorithm to be described in this section, we would like to obtain a neighbor of the current assignment $(d_v^1)_{v\in V}$ by solving a PCSP with 2 domain elements for every vertex. Let $\mathcal{P}_2$ denote this optimization problem. For every $v \in V$ we have to select two domain elements of the original problem. If for all $v \in V$ one domain element of $\mathcal{P}_2$ is given by d_v^1, then we guarantee that the new solution is at least as good as the current assignment. So, to construct $\mathcal{P}_2$ one additional domain element is necessary for every vertex. The second domain elements can be seen as a second solution s^2 that differs from s for all vertices. As a consequence, the construction of a neighborhood problem $\mathcal{P}_2$ for the current solution s is simply defined by the choice of a second solution s^2. This second solution can be generated in many different ways. We propose two different procedures for the generation of s' which will be called Best Neighbor, and Random.

Best Neighbor In the procedure Best Neighbor we generate the second assignment $(d_v^2)_{v\in V}$ by selecting for every vertex $v \in V$, the domain element d_v^2 that would increase the value of the assignment $(d_v^1)_{v\in V}$ as little as possible, i.e.,

$$d_v^2 = \arg \min_{d_v \in D_v \setminus \{d_v^1\}} \left\{ q(v, d_v) + \sum_{w \in N(v)} p(v, d_v, w, d_w^1) \right\}$$

This solution is in general not 1-optimal. Computational experiments have shown that the best results are obtained in case first 1-Opt is applied to $(d_v^2)_{v\in V}$ before the construction of $\mathcal{P}_2$. Based on the same experiments the solution $(d_v^1)_{v\in V}$ is also supposed to be 1-optimal.

Random In the procedure Random we just generate the second assignment $(d_v^2)_{v\in V}$ by randomly generating another domain element for every vertex. Again, we suppose that the first assignment $(d_v^1)_{v\in V}$ is 1-optimal, and that before $\mathcal{P}_2$ is constructed, we apply 1-Opt to the second solution $(d_v)_{v\in V}$ as well.

5.2.2 Computational Results

The two neighborhood functions described in the previous subsection are applied to the CALMA benchmark instances. The local search algorithms are implemented in C++ and run on a DEC 2100 A500MP workstation with 128Mb internal memory. We used the callable library of CPLEX 4.0 to solve the integer linear programming problems. Since all vertices have 2 domain elements, the separation of 3-cycle inequalities has been carried out by enumeration of the 4 different inequalities for every 3-cycle. Only 3-cycle inequalities are separated.

instance	1-Opt		IP-Opt - Best N.		IP-Opt - Random		GA [118]		best	cpu-time
	average	best	average	best	average	best	average	best	known	(sec/run)
CELAR 06	4,566	3,730	5,032	3,643	3,476	3,402	3,406	3,389	3,389	177
CELAR 07	2,502,232	1,404,410	5,314,174	2,495,854	380,173	344,093	343,604	343,593	343,592	345
CELAR 08	472	324	271	262	274	263	262	262	262	967
CELAR 09	18,714	15,740	15,573	15,571	15,571	15,571	15,571	15,571	15,571	146
CELAR 10	31,620	31,516	31,516	31,516	31,516	31,516	31,516	31,516	31,516	144
GRAPH 05	5,455	787	1,998	316	1,817	280	233	233	221	123
	-	-	-	-	238	221	-	-	-	250
GRAPH 06	11,826	8,410	6,805	4,297	6,213	4,373	4,133	4,133	4,124	371
	-	-	-	-	4,999	4,134	-	-	-	750
GRAPH 07	6,215	4,324	4,328	4,324	4,325	4,324	4,324	4,324	4,324	128
GRAPH 11	18,076	14,720	6,254	4,991	6,308	3,812	3,104	3,104	3,080	861
	-	-	-	-	4,045	3,093	-	-	-	1725
GRAPH 12	14,855	11,827	11,827	11,827	11,827	11,827	11,827	11,827	11,827	196
GRAPH 13	25,753	21,352	17,047	13,513	17,324	14,355	10,354	10,339	10,110	680
	-	-	-	-	14,341	10,155	-	-	-	1360

Table 5.1: Results local search algorithms based on integer programming results of Chapter 3. Framed values indicate the optimal solution.

Table 5.1 shows the results of the local search algorithms 1-Opt, IP-Opt - Best Neighbor, and IP-Opt - Random, as well as the results of the genetic algorithm of Kolen [118]. Like in [118], the population size of the genetic algorithm was set to 150. For the CELAR instances 10 new generations were computed, whereas for the GRAPH instances 15 generations were computed. For a fair comparison we have implemented the genetic algorithm with the same data structures as the local search algorithms. The time needed by the genetic algorithm has been taken as the maximum computation time for 10 runs of the local search algorithm, i.e. each run was limited by a tenth of the time needed by the genetic algorithm. Each run of the local search algorithm consists of (i) the generation of a random solution, (ii) the local improvement of the current solution according to the selected neighborhood function, and (iii) the disturbance of the best solution so far and re-application of the local improvement step. For the compared local search algorithms, Table 5.1 reports the average and best solution value obtained in 10 runs of the local search algorithms. The computation times in seconds per run are presented in the last column of table.

Table 5.1 shows that compared with 1-Opt, both IP-Opt algorithms provide a better best solution in most cases. Also the average values decrease substantially compared with the 1-Opt routine. The best results are obtained with the IP-Opt - Random. For the 4 instances with vertex penalties the algorithm outputs the optimal solution in almost all cases. For the other CELAR instances the best known / optimal solution is approached within 1%. For the more difficult GRAPH instances, the results are less satisfactory. Therefore, we applied the IP-Opt- Random algorithm with a time limit of two times the original limit. The results improve in this way substantially to optimal or near-optimal ones. In comparison with the genetic algorithm, the IP-Opt - Random algorithm generates competitive results.

5.3 Local Search and Tree Decomposition

In this section we describe a neighborhood structure that can take advantage of the tree decomposition approach described in Chapter 4. The computational results of Chapter 4 show that the dynamic programming algorithm can solve problems to optimality as long as the width of the tree decomposition is small. Therefore, we present in this section a neighborhood function in which the best neighbor is obtained by the solution of a PCSP on an induced subgraph with small treewidth. The neighborhood function is fairly general and can be seen as an extension of the 1-Opt function. In Section 5.3.1 the neighborhood function is defined, whereas Section 5.3.2 reports on preliminary computational results.

5.3.1 Neighborhood

A neighbor of a solution s in the 1-Opt-neighborhood differs from s for at most one vertex. The idea behind 1-Opt can be generalized to a neighborhood in which a neighbor of the solution differs for only a limited number of vertices. Since, the reassignment of domain elements to multiple vertices only leads to better results in case the vertices are connected, we can formalize the idea as follows. Let $\mathcal{V} = \{V_1, \ldots, V_n\}$ be a collection of n vertex subsets. Moreover, let the neighborhood function $\mathcal{N}$ be defined by $\mathcal{N}(s) = \cup_{i=1,\ldots,n}\mathcal{N}_i(s)$, where $\mathcal{N}_i(s)$ maps to all solutions $s' \in \mathcal{S}$ that differ from s only for vertices $v \in V_i$. For instance, $\mathcal{V} = \{\{v\} : v \in V\}$ is equivalent to the 1-Opt neighborhood. Another well known neighborhood is called 2-Opt and allows for reassignment of two adjacent vertices, i.e., $\mathcal{V} = E$.

For 1-Opt the question whether there exists a neighbor with smaller objective value can be solved by inspection. Also for 2-Opt the question can be answered efficiently. On the contrary, for the neighborhood with $\mathcal{V} = \{V\}$, the neighborhood problem is equivalent to the original problem and therefore $\mathcal{NP}$-hard. For general collections $\mathcal{V}$ the search whether there exists a neighbor with smaller objective value than the current solution can be carried out with the dynamic programming algorithm of Chapter 4. For every subset V_i, $i = 1, \ldots, n$, dynamic programming answers the question in time polynomial with respect to $|V_i|$ and $|D_v|$, but exponential with respect to the width of a tree decomposition of the induced subgraph $G[V_i]$. So, if $\omega(V_i)$ denotes the width of a tree decomposition for the induced subgraph $G[V_i]$, then $\omega(\mathcal{V}) = \max_{i=1,\ldots,n}\omega(V_i)$ determines the complexity of the algorithm. For 1-Opt and 2-Opt it is obvious that $\omega(\mathcal{V}) = 0$ and $\omega(\mathcal{V}) = 1$, respectively. However, the results of Chapter 4 show that as long as $\omega(\mathcal{V})$ is small the neighborhood can be applicable in practice. This validates the search for a collection $\mathcal{V}$ of subsets with the property that $\omega(\mathcal{V}) \leq K$, for some value K, e.g. $K = 4$ or $K = 5$.

The determination of subsets V_i with $\omega(V_i) \leq K$ is a question that certainly requires further investigation. For now, we restrict ourselves to 4 collections of subsets that can be determined easily: 1-Opt, 2-Opt, Cycle-Opt, and Clique-Opt:

1-Opt As already mentioned $\mathcal{V} = \{\{v\} : v \in V\}$. I.e., the assignment can be changed for only one vertex at a time. The value $\omega(\mathcal{V})$ equals 0 in this case.

2-Opt The assignment can be changed simultaneously for two adjacent vertices: $\mathcal{V} = E$. The value $\omega(\mathcal{V}) = 1$ in this case.

Cycle-Opt The assignment can be changed simultaneously for all vertices on a chordless cycle: $\mathcal{V} = \{C \subset V : G[C] \text{ is a chordless cycle}\}$. For a chordless cycle we can construct a tree decomposition with width 2, which implies that $\omega(\mathcal{V}) = 2$. In our preliminary experiments only cycles of size 3 are taken into account.

Clique-Opt The assignment can be changed simultaneously for all vertices that form a clique of size at most K: $\mathcal{V} = \{C \subset V : G[C] \text{ is a clique with } |C| \leq K\}$. For cliques the tree decomposition simply consists of a single node containing all vertices. As a consequence, $\omega(\mathcal{V}) = K - 1$. In our experiments we set $K = 4$.

Note that, $\mathcal{N}_{\text{1-Opt}} \subset \mathcal{N}_{\text{2-Opt}} \subset \mathcal{N}_{\text{Clique-Opt}}$, and $\mathcal{N}_{\text{3-Cycle-Opt}} \subset \mathcal{N}_{\text{Clique-Opt}}$.

5.3.2 Computational Results

We present preliminary computational results for the local search algorithms based on tree decomposition. Due to the property that heuristics based on tree decomposition are more time consuming than other algorithms, we limit ourselves to a demonstration of the potential power of the above described local search algorithms. For all CALMA benchmark instances we randomly generated 10 solutions. Next, we applied 1-Opt, 2-Opt, Cycle-Opt with 3-cycles, and Clique-Opt with $K \leq 4$ to the solutions. In Table 5.2 we report the average and best solution for each of these local improvement algorithms. Table 5.2 shows that for one of the easy instances (the ones with vertex penalties) the optimal solution is found after application of 2-Opt, and for two other instances after the application of Cycle-Opt. For the other instances, the best and average values decrease substantially after application of 2-Opt, Cycle-Opt, and Clique-Opt. From the results in Table 5.2, and taking into account that the algorithms are time consuming, we may conclude that it is worthwhile to consider the 2-Opt, Cycle-Opt, and Clique-Opt improvement heuristics as top-end heuristics to improve solutions obtained by other heuristics.

5.4 Concluding Remarks

In this chapter we have presented two local search algorithms that take advantage of the results of the previous chapters on exact solution methods. In Section 5.2 we presented a local search algorithm, that takes advantage of the polyhedral results of Chapter 3. The computational results, as well as the results of Kolen [118] show that (very) good solutions can be obtained in this way. In Section 5.3 we have proposed a local search framework where induced subgraphs are solved to optimality with the tree decomposition algorithm of Chapter 4. Preliminary computational results show the potential power of the algorithm. Local optimal solutions can substantially be improved by the use of the new neighborhood functions Cycle-Opt and Clique-Opt. Therefore, it is certainly worthwhile to study neighborhood functions with small $\omega(\mathcal{V})$ more thoroughly. A more efficient implementation of the tree decomposition algorithm for small subgraphs will reduce the computation times, and hence improve the results.

instance	$\|V\|$	$\|E\|$	1-Opt		2-Opt		Cycle-Opt		Clique-Opt		best known
			average	best	average	best	average	best	average	best	(optimal)
CELAR 06	100	350	13,054	10,084	7,645	4,576	6,515	3,646	5,961	3,646	3,389
CELAR 07	200	817	$1.7 \cdot 10^7$	$1.2 \cdot 10^7$	7,269,132	4,758,558	5,043,744	1,606,955	4,602,491	1,566,834	343,592
CELAR 08	458	1,655	944	754	654	418	511	388	463	356	262
CELAR 09	340	1,130	25,246	15,954	18,556	15,591	18,141	15,571	15,976	15,571	15,571
CELAR 10	340	1,130	32,514	31,516	31,526	31,516	31,516	31,516	31,516	31,516	31,516
GRAPH 05	100	416	14,382	9,381	6,077	3,047	5,012	1,924	4,299	1,924	221
GRAPH 06	200	843	27,268	20,598	19,887	16,188	17,258	12,301	14,076	7,810	4,123
GRAPH 07	200	843	12,012	6,517	4,811	4,364	4,654	4,324	4,654	4,324	4,324
GRAPH 11	340	1,425	42,687	34,219	26,021	19,232	21,124	13,310	15,150	7,783	3,080
GRAPH 12	340	1,255	27,754	23,973	15,516	13,042	12,324	11,827	12,227	11,827	11,827
GRAPH 13	458	1,877	61,319	58,660	40,566	30,519	33,408	27,569	27,363	22,557	10,110

Table 5.2: Results local search algorithms based on tree decomposition approach of Chapter 4. Framed values indicate the optimal solution.

6. Directions for Further Research and Concluding Remarks

In this concluding chapter, we briefly investigate several directions for further research on the minimum interference frequency assignment problem (MI-FAP). We close this thesis with some overall concluding remarks concerning frequency assignment problems.

6.1 Directions for Further Research

In this thesis we described two exact solution methods to solve the MI-FAP. We discussed integer programming and tree decomposition. However, in the literature several other exact solution techniques are available for combinatorial optimization problems. In this section, we briefly investigate a couple of promising other methods and relaxations. Successively, we discuss Benders Decomposition, Lagrangean Relaxation, and a Semi-Definite Programming relaxation. The last subsection is devoted to a new integer linear programming formulation for the MI-FAP.

6.1.1 Benders Decomposition[1]

In 1962, Benders [17] proposed a decomposition algorithm for mixed integer programs. For several specially structured mixed integer programs, with different classes of variables, successful application of Benders Decomposition is reported in the literature. Also the Partial Constraint Satisfaction formulation of the MI-FAP lends itself to the use of Benders Decomposition. For the PCSP, the variables can be divided in two classes: the vertex variables $y(v, d_v)$, and the edge variables $z(v, d_v, w, d_w)$. Application of Benders Decomposition to the formulation (3.1)-(3.5) (see page 53), results in a Benders master problem on the y variables, and a Benders subproblem that involves the dual of the edge constraints (3.3). The Benders subproblem decomposes along the edges of the constraint graph to the dual of a transportation problem for each edge, which implies that the subproblem can be solved in polynomial time. The master problem, however, cannot be solved efficiently in general.

[1] The discussion of Benders Decomposition is joint work with Olaf E. Flippo.

Moreover, a disadvantage of Benders Decomposition is that straightforward implementation of the algorithm leads in most cases to algorithms that converge very slowly: an exorbitant number of master problems have to be solved before optimality has been proved. Magnanti and Wong [136] discussed the problem of slow convergence in a general setting, and applied their ideas for acceleration to facility location. One of the aspects that cause the slow convergence is the degeneracy of the subproblem. Also the PCSP subproblems are renowned for their degeneracy. Magnanti and Wong introduced in this context the concept of *Pareto optimal cuts*. Without going into details, a Pareto optimal cut is a solution to the Benders subproblem, that dominates the other optimal solutions. Magnanti and Wong proved that such a Pareto optimal solution can be found by solving a second linear program, that selects among all optimal solutions of the Benders subproblem a Pareto optimal solution.

Limited computational experiments on the instances of Chapter 3 with $|D_v| = 2$ (Table 3.1, page 78), indicate that, although we add Pareto optimal cuts, the convergence rate of the algorithm is still very slow. Especially, solving the master problem by branch-and-bound becomes very time consuming after a couple of iterations. The addition of valid inequalities for either the original problem or the Benders master problem can probably resolve this problem. For the PCSP several classes of valid inequalities are derived in Chapter 3, which can be added to the formulation. However, adding these inequalities to the formulation leads to the loss of the decomposition of the subproblem to the dual of a transportation problem for every edge in the constraint graph. Therefore, derivation of valid inequalities for the Benders master problem should be considered as an alternative.

To conclude, the application of Benders Decomposition to solve PCSPs has not been a great success so far. Research in the direction of valid inequalities for the Benders master problem may result in an improved convergence rate. Moreover, the Pareto optimal cuts should be the topic of further research. Are Pareto optimal cuts the best we can do in case of the PCSP, or are there other possibilities that converge faster to the optimal solution?

6.1.2 Lagrangean Relaxation

Whereas Benders Decomposition takes advantage of the different classes of variables in a mixed integer program, Lagrangean Relaxation focuses on the different classes of constraints. Lagrangean relaxation was developed by Held and Karp [82, 83] in connection with the traveling salesman problem, and provides lower bounds for integer linear optimization problems. A survey on Lagrangean relaxation methods for solving integer linear programs is given by Fisher [55].

For the PCSP, either the constraints (3.2) or the constraints (3.3) can be relaxed through Lagrangean relaxation. Relaxation of the constraints (3.2) that guarantee the selection of

exactly one domain element does not lead to a Lagrangean relaxation problem that can be solved efficiently. Relaxation of the constraints (3.3), that connect the y and z variables, results in a Lagrangean relaxation problem that can be solved by its linear programming relaxation. This implies that the lower bound obtained through Langrangean relaxation is of the same quality as the linear programming relaxation of the original problem (3.1)-(3.5) (cf. Geoffrion [64]). By the results of Chapter 3 we know that for real-life instances this lower bound is very poor.

Nevertheless, relaxation of the constraints (3.3) is a direction that should be investigated further. In Chapter 4 we showed that for graphs with small treewidth the PCSP can be solved in polynomial time, whereas the linear programming relaxation of the PCSP does not equal the optimal solution. This implies that for subgraphs with small treewidth the PCSP can be solved in polynomial time. Hence, we relax only the constraints (3.3) involving a subset of the edges $E \setminus E'$. Then for fixed Langrangean multipliers λ, a PCSP $P'(\lambda)$ remains on the graph $G' = (V, E')$ as Lagrangean optimization problem. If the treewidth of G' is less than or equal a constant k, then $P'(\lambda)$ can be solved with the dynamic programming algorithm of Chapter 4. For $k > 1$, it holds that the linear programming relaxation of $P'(\lambda)$ does not equal the optimal solution in general, which implies that the lower bound obtained in this way can be better than the lower bound derived by the linear programming relaxation of the original problem P. A special case occurs whenever the remaining PCSP $P'(\lambda)$ can be solved with the constraint graph reduction techniques of Section 4.4.1.

Concluding, at first sight Lagrangean relaxation will not lead to good lower bounds. However, Lagrangean relaxation applied to only a subset of the edges may lead to lower bounds that are better than the linear programming relaxation. Therefore, it would be worthwhile to investigate the relation between the Lagrangean relaxation lower bounds and the treewidth k of the graph G'. The determination of the subgraph G' with treewidth k, however, introduces a (new) optimization problem: Select a maximum (weighted) subset E' of the edges E such that the subgraph (V, E') has treewidth at most k. For $k = 1$ a (maximum weighted) spanning tree solves the problem. For general k, graph-theoretic results concerning the characteristics of graphs with limited treewidth can hopefully be helpful to determine such subgraphs.

6.1.3 Semi-Definite Programming Relaxation

In the last decade, a substantial part of the mathematical programming research has been devoted to interior point methods for semi-definite programming (cf. Nesterov and Nemirovski [149]). In recent years not only results of theoretical nature were published, but also computational experience on solving combinatorial optimization problems with semi-definite programming was reported (see for instance Helmberg [84] and his references).

The MI-FAP (or PCSP) can also be formulated as a semi-definite program. Let $y = [y(v,d_v)]_{v\in V,d_v\in D_v}$ be the vector containing all y variables. Moreover, let $Z = yy^T = [z(v,d_v,w,d_w)]_{v,w\in V,d_v\in D_v,d_w\in D_w}$ be the matrix with z variables. Note that, Z also contains variables for non-adjacent vertices. Finally, let

$$X = \begin{pmatrix} 1 \\ y \end{pmatrix} \begin{pmatrix} 1 & y^T \end{pmatrix} = \begin{pmatrix} 1 & y^T \\ y & Z \end{pmatrix}$$

By definition the matrix X is positive semi-definite. Moreover, $\text{rank}(X) = 1$. A feasible solution has to satisfy $y(v,d_v) \in \{0,1\}$, which is equivalent with $y(v,d_v) = (y(v,d_v))^2$, or in matrix notation $x_{i1} = x_{ii}$ for all i. Altogether, the MI-FAP (or PCSP) reads

$$\min \quad \text{tr}\left(\tfrac{1}{2}\begin{pmatrix} 0 & q^T \\ q & P \end{pmatrix} X\right) \tag{6.1}$$

$$\text{s.t.} \quad x_{i1} = x_{ii} \qquad \forall i \tag{6.2}$$

$$\sum_{d_v\in D_v} y(v,d_v) = 1 \qquad \forall v \in V \tag{6.3}$$

$$\text{rank}(X) = 1 \tag{6.4}$$

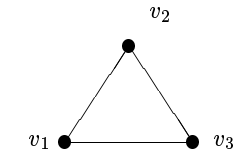
$$X \succeq 0 \tag{6.5}$$

where $\text{tr}(.)$ denotes the trace of a matrix, and $X \succeq 0$ denotes that the matrix X has to be positive semi-definite. The objective (6.1) is the sum product of the vertex and edge penalties and the matrix X. The constraints (6.2) model that the y variables have to be integral, whereas (6.3) model the assignment of exactly one frequency to every vertex. The formulation is completed with the constraints (6.4) and (6.5), that model the rank requirement, and the positive semi-definiteness of the matrix X, respectively.

Relaxation of (6.1)-(6.5) by removing the rank constraint (6.4) provides a semi-definite program, that can be solved within ϵ of optimal in polynomial time with interior point methods [149]. The semi-definite programming relaxation can be better / worse than the linear programming relaxation. For instance, negative variables $z(v,d_v,w,d_w)$ are not forbidden by the semi-definite relaxation. To improve the relaxation $z(v,d_v,w,d_w) \geq 0$ can be added to (6.1)-(6.3), (6.5). Also constraints like

$$\sum_{d_v\in D_v} \sum_{d_w\in D_w} z(v,d_v,w,d_w) = 1 \tag{6.6}$$

for $\{v,w\} \in E$ can improve the relaxation. Finally, valid inequalities like those of the integer linear programming formulation of Chapter 3 can be added to the formulation



$G = (V, E)$

$D_v = \{a_v, b_v\}$ $\quad \forall v \in V$

$q(v, d_v) = 0$ $\quad \forall v \in V, d_v \in D_v$

$p(v, d_v, w, d_w)$:

$\{v_1, v_2\}$	a_{v_2}	b_{v_2}
a_{v_1}	0	1
b_{v_1}	1	0

$\{v_1, v_3\}$	a_{v_3}	b_{v_3}
a_{v_1}	1	0
b_{v_1}	0	1

$\{v_2, v_3\}$	a_{v_3}	b_{v_3}
a_{v_2}	0	1
b_{v_2}	1	0

Figure 6.1: Example of PCSP with good semi-definite relaxation compared to linear programming relaxation

to improve the lower bound. An example of a PCSP in which the semi-definite relaxation is better than the linear programming relaxation is given by Figure 6.1. Using the semi-definite programming solver SeDuMi [177], Sturm [178] reported a semi-definite relaxation bound of 0.75, whereas the linear programming relaxation equals 0 and the optimal solution is 1.

To conclude, the semi-definite relaxation is a promising direction for further research. With the ongoing development of software to solve semi-definite programs, the positive semi-definite relaxation of (6.1)-(6.5) seems to be an attractive alternative for the linear programming relaxation of (3.1)-(3.5) within the near future. Probably the most important obstacle will be the size of the matrix X.

6.1.4 Frequency Assignment Formulation

The approaches described in the previous chapters and sections are based on the formulation of the MI-FAP as a Partial Constraint Satisfaction Problem. In this section we propose a new integer linear programming formulation that is more specialized for the MI-FAP. The formulation has fewer variables and constraints than the PCSP formulation, and therefore hopefully performs better in practice. Although the new formulation is not inspired by the formulation of Borndörfer et al. [24], it can be seen as a refinement and extension of their model.

In most MI-FAPs, given an edge $\{v, w\} \in E$, the penalty given by the domain elements $d_v \in D_v$ and $d_w \in D_w$ can be specified by

$$p(v, d_v, w, d_w) = \begin{cases} p_{vw} & \text{if } |d_v - d_w| < \delta_{vw} \\ 0 & \text{otherwise} \end{cases}$$

where p_{vw} is an edge-dependent constant, and δ_{vw} is a constant specifying the minimum reuse distance between the frequencies of v and w. So, depending on the edge $\{v, w\} \in E$

and the distance between the frequencies, the penalty is either zero or a constant value. In fact, the distance in between the frequencies can be divided in 3 intervals (i) $(d_v - d_w) \leq -\delta_{vw}$, (ii) $-\delta_{vw} < (d_v - d_w) < \delta_{vw}$, and (iii) $(d_v - d_w) \geq \delta_{vw}$. More general, we assume that for every edge $\{v, w\} \in E$ we can specify n_{vw} intervals $[l^i_{vw}, u^i_{vw}]$ for the difference between the frequencies. These intervals cover all possible differences (i.e., $l^1_{vw} = -\infty$, $u^{n_{vw}}_{vw} = \infty$, and $l^i_{vw} = u^{i-1}_{vw} + 1$, for $i = 2, \ldots, n_{vw}$). For every interval the penalty is fixed:

$$p(v, d_v, w, d_w) = p^i_{vw} \qquad \text{if } l^i_{vw} \leq d_v - d_w \leq u^i_{vw}, i = 1, \ldots, n_{vw}$$

Note that $p(v, d_v, w, d_w)$ need not be symmetric in general. The MI-FAP can be modeled as an integer linear program that takes advantage of this penalty structure. For every edge $\{v, w\} \in E$ and every $i = 1, \ldots, n_{vw}$ we introduce a binary variable x^i_{vw}:

$$x^i_{vw} = \begin{cases} 1 & \text{if we assign frequencies } d_v \in D_v,\ d_w \in D_w,\ l^i_{vw} \leq d_v - d_w \leq u^i_{vw} \\ 0 & \text{otherwise} \end{cases}$$

Moreover, we introduce binary variables y^j_v for all vertices $v \in V$, $j \in \{1, \ldots, |D_v|\}$

$$y^j_v = \begin{cases} 1 & \text{if } j^{th} \text{ element of } D_v \text{ is assigned to } v \in V \\ 0 & \text{otherwise} \end{cases}$$

Finally, we need integer variables d_v denoting the selected frequency for all vertices $v \in V$. So, d_v is not an index this time, but a variable. Then an integer linear programming formulation reads

$$\min \quad \sum_{\{v,w\} \in E} \sum_{i=1}^{n_{vw}} p^i_{vw} x^i_{vw} + \sum_{v \in V} \sum_{j=1}^{|D_v|} q^j_v y^j_v \tag{6.7}$$

$$\text{s.t.} \quad \sum_{i=1}^{n_{vw}} x^i_{vw} = 1 \qquad \forall \{v, w\} \in E \tag{6.8}$$

$$\sum_{j=1}^{|D_v|} y^j_v = 1 \qquad \forall v \in V \tag{6.9}$$

$$d_v = \sum_{j=1}^{|D_v|} d^j_v y^j_v \qquad \forall v \in V \tag{6.10}$$

$$d_w \geq d_v - \sum_{i=1}^{n_{vw}} u^i_{vw} x^i_{vw} \qquad \forall \{v, w\} \in E \tag{6.11}$$

$$d_v \geq d_w + \sum_{i=1}^{n_{vw}} l^i_{vw} x^i_{vw} \qquad \forall \{v, w\} \in E \tag{6.12}$$

$$d_v \in \mathbb{Z} \qquad \forall v \in V \tag{6.13}$$

$$x^i_{vw} \in \{0,1\} \qquad \forall \{v,w\} \in E, i = 1,\ldots,n_{vw} \tag{6.14}$$

$$y^j_v \in \{0,1\} \qquad \forall v \in V, j = 1,\ldots,|D_v| \tag{6.15}$$

Here, q^j_v and d^j_v denote respectively the vertex penalty and the value (frequency) corresponding to the j^{th} domain element of D_v, $1 \leq j \leq |D_v|$. The objective (6.7) equals the sum of edge and vertex penalties. Constraints (6.8) model the fact that for an edge $\{v,w\} \in E$ the difference between the assigned frequencies is contained in exactly one interval $[l^i_{vw}, u^i_{vw}]$. Constraints (6.9) enforce that exactly one domain element is selected, whereas the selected frequency is given by (6.10). Given a vector x satisfying (6.8), the assignment $(d_v)_{v \in V}$ has to satisfy $l^i_{vw} \leq d_v - d_w \leq u^i_{vw}$ if $x^i_{vw} = 1$ for all $\{v,w\} \in E$. This restriction can be split up in two restrictions $d_w \geq d_v - u^i_{vw}$, and $d_v \geq d_w + l^i_{vw}$ for each $\{v,w\} \in E$. Combined with the variables x^i_{vw} these restrictions are modeled by the constraints (6.11) and (6.12), respectively.

Compared to the Partial Constraint Satisfaction formulation (3.1)-(3.5), the number of constraints and variables is reduced substantially. The number of binary variables however is increased. In Table 6.1, we have compared the number of variables and constraints for some of the CALMA benchmark instances.

instance	number of variables			number of constraints	
	(3.1)-(3.5)		(6.7)-(6.15)	(3.1)-(3.5)	(6.7)-(6.15)
	continuous	binary	binary		
CELAR 06	585,914	4,010	7,876	28,628	4,366
CELAR 07	1,331,048	7,976	16,371	65,928	9,395
CELAR 08	2,518,664	18,100	34,874	128,886	19,064
GRAPH 11	2,032,792	12,820	23,211	107,688	12,631
GRAPH 13	2,798,400	17,588	32,949	145,034	17,651

Table 6.1: Number of variables and constraints for the different integer programming formulations

Summarized, the formulation (6.7)-(6.15) seems to be a profitable alternative to the partial constraint satisfaction formulation of Chapter 3. Directions of further research include study of the polyhedral structure of the corresponding polytope (cf. Borndörfer et al. [24]), and further improvements of the formulation in special cases.

6.2 Conclusions

In this Ph.D. thesis, we have discussed models and algorithms for the Frequency Assignment Problem. After an introduction of the topic in Chapter 1 we first presented a survey concerning the different approaches presented in the recent literature in Chapter 2. Four different models can be distinguished: minimum order frequency assignment, minimum span frequency assignment, minimum blocking frequency assignment, and minimum interference frequency assignment. For each of these models we compared the wide variety of approaches as far as possible on the same sets of instances.

In the sequel of the thesis we have concentrated on the minimum interference frequency assignment problem (MI-FAP). Successful lower bounds and exact solution techniques are rarely known for this problem. Therefore, we applied two exact solution methods to the problem in the Chapters 3 and 4, respectively. Other lower bounding and exact solution techniques were discussed in Chapter 6. In Chapter 3 we formulated the MI-FAP as a partial constraint satisfaction problem, and studied this problem from a polyhedral point of view. We derived two general lifting theorems, and two classes of facet defining valid inequalities. Problems with small domains can be solved effectively with these inequality in a cutting plane algorithm. For real-life instances, however, this method is not powerful enough to solve the problems. However, in Chapter 5 we showed that the polyhedral results can be used in a heuristic that provides fairly good solutions to the benchmark instances. Moreover, Kolen [118] showed that a genetic algorithm, which uses the polyhedral results of Chapter 3, outperforms all other proposed heuristics on the benchmark instances. The results of Chapter 4 show that for 7 of the 11 instances the solutions obtained in this way are optimal.

In Chapter 4 we exploited the graph structure of the problem in order to solve the benchmark instances from the CALMA project, or second best obtain lower bounds for them. The method is based on a tree decomposition of the constraint graph. Given a tree decomposition with limited width, the MI-FAP (or PCSP) can be solved through dynamic programming in polynomial time. However, the algorithm is exponential in the width of the tree decomposition, which explains that additional reduction techniques are necessary to solve several benchmark instances. Successful application of this technique is, however, limited to the smaller and less difficult instances. For the larger and more difficult instances the approach is extended to an iterative algorithm that provides a series of non-decreasing lower bounds. In this way, we obtained the first non-trivial lower bounds for the remaining unsolved instances. The iterative algorithm can also be combined with the integer programming techniques of Chapter 3. This combination resulted in an improve of the lower bounds for the 3 most difficult instances. Finally, in Chapter 5 we showed that the techniques of Chapter 4 can be used within a local search framework as well. Preliminary computational results indicate that the proposed neighborhood is substantially better than less sophisticated neighborhoods.

In this chapter we have discussed several other exact methods to solve the MI-FAP. Especially, Lagrangean relaxation combined with tree decomposition (Section 6.1.2), and the new integer programming formulation of Section 6.1.4 are worthwhile research directions.

Summarized, in this Ph.D. thesis, we presented a survey on frequency assignment, we solved 7 out of the 11 CALMA MI-FAP benchmark instances to optimality, and derived the first non-trivial lower bounds for the other instances (see Table 6.2). Moreover, we presented two new heuristics, based on the exact methods of integer programming and tree decomposition, respectively. Finally, we discussed four other solution methods that can be the topic of further research.

instance	previous lower bound	new lower bound	upper bound	gap closed by
CELAR 06	3,389	3,389	3,389	-
CELAR 07	5	300,000	343,592	87.3%
CELAR 08	0	150	262	57.3%
CELAR 09	14,969	15,571	15,571	100.0%
CELAR 10	31,204	31,516	31,516	100.0%
GRAPH 05	0	221	221	100.0%
GRAPH 06	0	4,123	4,123	100.0%
GRAPH 07	0	4,324	4,324	100.0%
GRAPH 11	0	3,016	3,080	97.9%
GRAPH 12	0	11,827	11,827	100.0%
GRAPH 13	0	9,925	10,110	98.2%

Table 6.2: Lower and upper bounds for the Minimum Interference benchmark instances of the CALMA project. Framed values indicate the optimal value.

Bibliography

Bibliografie

[1] K.I. Aardal, A. Hipolito, C.P.M. van Hoesel, and B. Jansen. A branch-and-cut algorithm for the frequency assignment problem. Research Memorandum 96/011, Maastricht University, 1996. Available at http://www.unimaas.nl/~svhoesel/.

[2] K.I. Aardal and C.P.M. van Hoesel. Polyhedral techniques in combinatorial optimization. I: Theory. *Statistica Neerlandica*, 50:3–26, 1996.

[3] K.I. Aardal and C.P.M. van Hoesel. Polyhedral techniques in combinatorial optimization II: Applications and computations. *Statistica Neerlandica*, 53:131–177, 1999.

[4] K.I. Aardal, C.P.M. van Hoesel, A.M.C.A. Koster, C. Mannino, and A. Sassano. Models and solution techniques for the frequency assignment problem. In preparation, Maastricht University, 1999.

[5] K.I. Aardal, C.A.J. Hurkens, J.K. Lenstra, and S.R. Tiourine. Algorithms for frequency assignment problems (extended abstract). *CWI Quarterly*, 9:1–8, 1996.

[6] K.I. Aardal, C.A.J. Hurkens, J.K. Lenstra, and S.R. Tiourine. Algorithms for frequency assignment problems. Technical report, Technische Universiteit Eindhoven, 1999.

[7] E.H.L. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines.* John Wiley and Sons, Chichester, 1989.

[8] E.H.L. Aarts and J.K. Lenstra, editors. *Local Search in Combinatorial Optimization.* Wiley, Chichester, 1997.

[9] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, New Jersey, 1993.

[10] F.S. Al-Khaled. Optimal radio channel assignment through the new binary dynamic simulated annealing algorithm. *International Journal of Communication Systems*, 11:327–336, 1998.

[11] S.M. Allen, D.H. Smith, and S. Hurley. Lower bounding techniques for frequency assignment. *Discrete Mathematics*, 197/198:41–52, 1999.

[12] L.G. Anderson. A simulation study of some dynamic channel assignment algorithms in a high capacity mobile telecommunications system. *IEEE Transactions on Communications*, 21:1294–1301, 1973.

[13] S. Arnborg, D.G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic and Discrete Methods*, 8:277–284, 1987.

[14] M. Balinski. On a selection problem. *Management Science*, 17:230–231, 1970.

[15] F. Barahona. A solvable case of quadratic 0-1 programming. *Discrete Applied Mathematics*, 13:23–26, 1986.

[16] J. Bater, P. Jeavons, and D. Cohen. Are there optimal reuse distance constraints for FAPs with random Tx placements? Technical Report CSD-TR-98-01, Royal Holloway University of London, 1998.

[17] J.F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[18] H.P. van Benthem. GRAPH generating radio link frequency assignment problems heuristically. Master's thesis, Delft University of Technology, 1995.

[19] D. Bienstock. Private communication, 1998.

[20] M. Biró, M. Hujter, and Zs. Tuza. Precoloring extensions. I. interval graphs. *Discrete Mathematics*, 100:267–279, 1992.

[21] H.L. Bodlaender. A tourist guide through treewidth. *Acta Cybernetica*, 11(1-2):1–21, 1993.

[22] H.L. Bodlaender. A linear time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

[23] R. Borndörfer, A. Eisenblätter, M. Grötschel, and A. Martin. Frequency assignment in cellular phone networks. *Annals of Operations Research*, 76:73–94, 1998.

[24] R. Borndörfer, A. Eisenblätter, M. Grötschel, and A. Martin. The orientation model for frequency assignment problems. Technical Report TR 98-01, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 1998.

[25] A. Bouju, J.F. Boyce, C.H.D. Dimitropoulos, G. Vom Scheidt, and J.G. Taylor. Tabu search for the radio links frequency assignment problem. In *Applied Decision Technologies (ADT'95)*, London, 1995.

[26] A. Bouju, J.F. Boyce, C.H.D. Dimitropoulos, G. Vom Scheidt, J.G. Taylor, A. Likas, G. Papageorgiou, and A. Stafylopatis. Intellegent search for the radio links frequency assignment problem. In *Int. Conf. For Digital Signal Processing (DSP'95)*, Limassol, Cypres, 1995.

[27] F. Box. A heuristic technique for assigning frequencies to mobile radio nets. *IEEE Transactions on Vehicular Technology*, 27:57–74, 1978.

[28] D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22:251–256, 1979.

[29] Encyclopaedia Britannica 98. Multimedia Edition, 1998.

[30] B. Brockmüller, O. Günlük, and L.A. Wolsey. Designing private line networks - polyhedral analysis and computation. Discussion Paper 9647, Center for Operations Research and Econometrics, October 1996.

[31] B. Cabon, S. De Givry, L. Lobjois, T. Schiex, and J.P. Warners. Benchmarks problems: Radio link frequency assignment. *Constraints*, 4:79–89, 1999.

[32] EUCLID CALMA project. Publications available at FTP Site: `ftp.win.tue.nl`, Directory `/pub/techreports/CALMA`, 1995.

[33] D.J. Castelino, S. Hurley, and N.M. Stephens. A tabu search algorithm for frequency assignment. *Annals of Operations Research*, 63:301–319, 1996.

[34] D.J. Castelino and N.M. Stephens. Tabu thresholding for the frequency assignment problem. In I.H. Osman and J.P. Kelly, editors, *Meta-Heuristics. Theory and Applications*, chapter 22, pages 343–359. Kluwer Academic Publishers, 1996.

[35] D.J. Castelino and N.M. Stephens. A surrogate constraint tabu thresholding implementation for the frequency assignment problem. *Annals of Operations Research*, 86:259–270, 1999.

[36] K.-N. Chang and S. Kim. Channel allocation in cellular radio networks. *Computers and Operations Research*, 24:849–860, 1997.

[37] W. Cook. Private communication, 1997.

[38] D. Costa. On the use of some known methods for t-colourings of graphs. *Annals of Operations Research*, 41:343–358, 1993.

[39] M.B. Cozzens and F.S. Roberts. T-colorings of graphs and the channel assignment problem. *Congressus Numerantium*, 35:191–208, 1982.

[40] C. Crisan and H. Mühlenbein. The breeder genetic algorithm for frequency assignment. *Lecture Notes in Computer Science*, 1498:897–906, 1998.

[41] C. Crisan and H. Mühlenbein. The frequency assginment problem: A look at the performance of evolutionary search. *Lecture Notes in Computer Science*, 1363:263–274, 1998.

[42] M. Cuppini. A genetic algorithm for channel assignment problems. *European Transactions on telecommunications and related technologies*, 5:285–294, 1994.

[43] E. Dahlhaus, D.S. Johnson, C.H. Papadimitriou, P.D. Seymour, and M. Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23:864–894, 1994.

[44] J.E. Dayhoff. *Neural Networks Architectures: An Introduction*. Van Nostrand Reinhold, New York, 1990.

[45] R. Dorne and J.-K. Hao. An evolutionary approach for frequency assignment in cellular radio networks. In *IEEE Int. Conference on Evolutionary Computing*, Perth, Australia, 1995.

[46] R. Dorne and J.-K. Hao. Constraint handling in evolutionary search: A case study of the frequency assignment. *Lecture Notes in Computer Science*, 1141:801–810, 1996.

[47] R. Dorne and J.-K. Hao. A new genetic local search algorithm for graph coloring. *Lecture Notes in Computer Science*, 1498:745–754, 1998.

[48] N. Dunkin and S.M. Allen. Frequency assignment problems: Representations and solutions. Technical Report CSD-TR-97-14, Royal Holloway, University of London, 1997. Available at http://www.dcs.rhbnc.ac.uk/research/constraints/publications/-index.shtml.

[49] N. Dunkin, J. Bater, P. Jeavons, and D. Cohen. Towards high order constraint representations for the frequency assignment problem. Technical Report CSD-TR-98-05, Royal Holloway University of London, 1998. Available at http://www.dcs.rhbnc.ac.uk/research/constraints/publications/index.shtml.

[50] M. Duque-Antón, D. Kunz, and B. Rüber. Channel assignment for cellular radio using simulated annealing. *IEEE Transactions on Vehicular Technology*, 42:14–21, 1993.

[51] EMC world cellular database, 1999. http://www.emc-database.com/.

[52] Microsoft Encarta 97 Encyclopedia. CD-ROM, 1997.

[53] P. Erdös, A.L. Rubin, and H. Taylor. Choosability in graphs. *Congressus Numerantium*, 26:125–157, 1979.

[54] M. Fischetti, C. Lepschy, G. Minerva, G. Romanin Jacur, and E. Toto. Frequency assignment in mobile radio systems using branch-and-cut techniques. Technical report, Universita di Padova, 1998.

[55] M.L. Fisher. The lagrangian relaxation method for solving integer programming problems. *Management Science*, 27:1–18, 1981.

[56] O.E. Flippo, A.W.J. Kolen, A.M.C.A. Koster, and R.L.M.J. van de Leensel. A dynamic programming algorithm for the local access telecommunication network expansion problem. *European Journal of Operational Research*, to appear, 1999.

[57] E.C. Freuder and R.J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58:21–70, 1992.

[58] N. Funabiki and S. Nishikawa. A gradual neural-network approach for frequency assignment in satellite communication systems. *IEEE Transactions on Neural Networks*, 8:1359–1370, 1997.

[59] N. Funabiki and Y. Takefuji. A neural network parallel algorithm for channel assignment problems in cellular radio networks. *IEEE Transactions on Vehicular Technology*, 41:430–437, 1992.

[60] A. Gamst. Some lower bounds for a class of frequency assignment problems. *IEEE Transactions on Vehicular Technology*, 35:8–14, 1986.

[61] M.R. Garey and D.S. Johnson. The complexity of near-optimal graph coloring. *Journal of the ACM*, 23:43–49, 1976.

[62] M.R. Garey and D.S. Johnson. *Computers and intractability: a guide to the Theory of $\mathcal{NP}$-Completeness*. Freeman and Company, N.Y., 1979.

[63] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[64] A.M. Geoffrion. Lagrangian relaxation and its uses in integer programming. *Mathematical Programming Study*, 2:82–114, 1974.

[65] A.I. Giortzis and L.F. Turner. Application of mathematical programming to the fixed channel assignment problem in mobile radio networks. *IEE Proceedings - Communications*, 144:257–264, 1997.

[66] S. De Givry, G. Verfaillie, and T. Schiex. Bounding the optimum of constraint optimization problems. In *Proceedings of the 3rd International Conference on Principles and Practice of Constraint Programming (CP-97)*, 1997.

[67] F. Glover. A multiphase-dual algorithm for the zero-one integer programming problem. *Operations Research*, 13:879–919, 1965.

[68] F. Glover. Heuristics for integer programming using surrogate constraints. *Decision Science*, 8:156–166, 1977.

[69] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13:543–549, 1986.

[70] F. Glover. Tabu search - part I. *ORSA Journal on Computing*, 1:190–206, 1989.

[71] F. Glover. Tabu search - part II. *ORSA Journal on Computing*, 2:4–32, 1990.

[72] F. Glover. Tabu thresholding: Improved search by nonmonotonic trajectories. *ORSA Journal on Computing*, 7:426–442, 1995.

[73] F. Glover and M. Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, 1997.

[74] D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, Ma., 1989.

[75] R.E. Gomory and T.C. Hu. Multi-terminal network flows. *Journal of SIAM*, 9:551–570, 1961.

[76] J.R. Griggs and D.D.-F. Liu. The channel assignment problem for mutually adjacent sites. *Journal of Combinatorial Theory. Series A*, 68:169–183, 1994.

[77] GSM Association annual report 1998, 1998. Available at http://www.gsmworld.com.

[78] W.K. Hale. Frequency assignment: Theory and applications. *Proceedings of the IEEE*, 68:1497–1514, 1980.

[79] P. Hansen. Methods of nonlinear 0-1 programming. *Annals of Discrete Mathematics*, 5:53–70, 1979.

[80] J.-K. Hao and R. Dorne. Study of genetic search for the frequency assignment problem. *Lecture Notes in Computer Science*, 1063:333–344, 1996.

[81] J.-K. Hao, R. Dorne, and P. Galinier. Tabu search for frequency assignment in moblie radio networks. *Journal of Heuristics*, 4:47–62, 1998.

[82] M. Held and R.M. Karp. The traveling-salesman problem and minimum spanning trees. *Operations Research*, 18:1138–1162, 1970.

[83] M. Held and R.M. Karp. The traveling-salesman problem and minimum spanning trees: Part II. *Mathematical Programming*, 1:6–25, 1971.

[84] C. Helmberg. Fixing variables in semidefinite relaxations. Technical Report SC 96-43, ZIB, Berlin, 1996.

[85] J. van den Heuvel, R.A. Leese, and M.A. Shepherd. Graph labelling and radio channel assignment. *Journal of Graph Theory*, 29:263–284, 1998.

[86] C.P.M. van Hoesel, A.M.C.A. Koster, R.L.M.J. van de Leensel, and M.W.P. Savelsbergh. Polyhedral results for the edge capacity polytope. Technical report, Maastricht University, 1999.

[87] J.H. Holland. *Adaption in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, 1975.

[88] J.J. Hopfield and D.W. Tank. Neural computation of decisions in optimization problems. *Biological Cybernetics*, 52:141–152, 1985.

[89] J.J. Hopfield and D.W. Tank. Computing with neural circuits: A model. *Science*, 233:625–633, 1986.

[90] T.C. Hu. *Integer Programming and Network Flows.* Addison-Wesley Publishing Co., Reading, MA, 1969.

[91] C.A.J. Hurkens and S.R. Tiourine. Upper and lower bounding techniques for frequency assignment problems. Technical Report COSOR 95-34, Eindhoven University of Technology, 1995. Available at http://www.win.tue.nl/math/bs/comb_opt/-hurkens/calma.html.

[92] S. Hurley, D.H. Smith, and S.U. Thiel. FASoft: A system for discrete channel frequency assignment. *Radio Science*, 32:1921–1939, 1997.

[93] Iridium inc., web-site: Http://www.iridium.com, 1999.

[94] K. Jansen and P. Scheffler. Generalized coloring for tree-like graphs. *Lecture Notes in Computer Science*, 657:50–59, 1993.

[95] J. Janssen and K. Kilakos. An optimal solution to the "Philadelphia" channel assignment problem". Technical Report CDAM-96-16, London School of Economics, 1996.

[96] J. Janssen and K. Kilakos. Polyhedral analysis of channel assignment problems: (i) tours. Technical Report CDAM-96-17, London School of Economics, 1996.

[97] J. Janssen and K. Kilakos. Tile covers, closed tours and the radio spectrum. Technical Report CDAM-97-08, London School of Economics, 1997.

[98] J. Janssen, K. Kilakos, and O. Marcotte. Fixed preference channel assignment for cellular telephone systems. *IEEE Transactions on Vehicular Technology*, 48:533–541, 1999.

[99] B. Jaumard, O. Marcotte, and C. Meyer. Estimation of the quality of cellular networks using column generation techniques. Technical report, Ecole Polytechnique de Montréal, January 1998.

[100] B. Jaumard, O. Marcotte, and C. Meyer. Mathematical models and exact methods for channel assignment in cellular networks. In B. Sansáo and P. Soriano, editors, *Telecommunications Network Planning*, chapter 13, pages 239–255. Kluwer Academic Publishers, Boston, 1999.

[101] B. Jaumard, O. Marcotte, C. Meyer, and T. Vovor. Comparison of column generation models for channel assignment in cellular networks. Technical report, Ecole Polytechnique de Montréal, October 1998.

[102] B. Jaumard and T. Vovor. A column generation approach for the exact solution of channel assignment problems. Technical report, Ecole Polytechnique de Montréal, July 1998.

[103] P. Jeavons, N. Dunkin, and J. Bater. Why higher order constraints are necessary to model frequency assignment problems. In *Proceedings of ECAI 98*. John Wiley & Sons, Ltd., 1998.

[104] P.K. Johri. An insight into dynamic channel assignment in cellular mobile communications systems. *European Journal of Operational Research*, 74:70–77, 1994.

[105] A.P. Kamath, N.K. Karmarkar, K.G. Ramakrishnan, and M.G.C. Resende. Computational experience with an interior poitn algorithm on the satisfiability problem. *Annals of Operations Research*, 25:43–58, 1990.

[106] A. Kapsalis, P. Chardaire, V.J. Rayward-Smith, and G.D. Smith. The radio link frequency assignment problem: A case study using genetic algorithms. *Lecture Notes on Computer Science*, 993:117–131, 1995.

[107] A. Kapsalis, V.J. Rayward-Smith, and G.D. Smith. Using genetic algorithms to solve the radio link frequency assignment problem. In D.W. Pearson, N.C. Steele, and R.F. Albrecht, editors, *Proceedings of the Second International Conference on Artificial Neural Networks and Genetic Algorithms*. Springer Verlag, 1995.

[108] N.K. Karmarkar. An interior-point approach to NP-complete problems - part I. *Contemporary Mathematics*, 114:297–308, 1990.

[109] N.K. Karmarkar, M.G.C. Resende, and K.G. Ramakrishnan. An interior point algorithm to solve computationally difficult set covering problems. *Mathematical Programming*, 52:597–618, 1991.

[110] R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, New York, 1972.

[111] I. Katzela and M. Naghshineh. Channel assignment schemes for cellular mobile telecommunication systems. *Personal Communications Magazine*, 1996.

[112] M.G. Kazantzakis, P.P. Demestichas, and M.E. Anagnostou. Optimum frequency reuse in mobile telephone systems. *International Journal of Communications Systems*, 8:185–190, 1995.

[113] J.-S. Kim, S. Park, P. Dowd, and N. Nasrabadi. Channel assignment in cellular radio using genetic algorithms. *Wireless Personal Communications*, 3:273–286, 1996.

[114] S. Kim and S.-L. Kim. A two-phase algorithm for frequency assignment in cellular mobile systems. *IEEE Transactions on Vehicular Technology*, 43:542–548, 1994.

[115] S. Kirkpatrick, C.D. Gelatt, and M.D. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[116] D. Klabjan, G.L. Nemhauser, and C. Tovey. The complexity of cover inequality separation. *Operations Research Letters*, 23:35–40, 1998.

[117] A. Knälmann and A. Quellmalz. Solving the frequency assignment problem with simulated annealing. *IEE conference publication*, 396:233–240, 1994.

[118] A.W.J. Kolen. A genetic algorithm for frequency assignment. Technical report, Maastricht University, 1999. Available at http://www.unimaas.nl/~akolen/.

[119] A.W.J. Kolen. Private communication, 1999.

[120] A.W.J. Kolen, C.P.M. van Hoesel, and R. van der Wal. A constraint satisfaction approach to the radio link frequency assignment problem. Technical Report 2.2.2, EUCLID CALMA project, 1994.

[121] A.M.C.A. Koster, C.P.M. van Hoesel, and A.W.J. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3-5):89–97, 1998.

[122] A.M.C.A. Koster, C.P.M. van Hoesel, and A.W.J. Kolen. Optimal solutions for a frequency assignment problem via tree-decomposition. In *Lecture Notes in Computer Science*. Graph-Theoretic Concepts in Computer Science (WG '99), Springer-Verlag, 1999.

[123] A.M.C.A. Koster, C.P.M. van Hoesel, and A.W.J. Kolen. Solving frequency assignment problems via tree-decomposition. Technical Report RM 99/011, Maastricht University, 1999. Available at http://www.unimaas.nl/~akoster/.

[124] A.M.C.A. Koster, C.P.M. van Hoesel, and A.W.J. Kolen. Solving frequency assignment problems via tree-decomposition. *Electronic Notes on Discrete Mathematics*, 3, 1999. Available at http://www.elsevier.nl/locate/endm/.

[125] M. Kubale. Interval vertex-coloring of a graph with forbidden colors. *Discrete Mathematics*, 74:125–136, 1989.

[126] M. Kubale. Some results concerning the complexity of restricted colorings of graphs. *Discrete Applied Mathematics*, 36:35–46, 1992.

[127] V. Kumar. Algorithms for constraint satisfaction problems: A survey. *AI Magazine*, 13(1):32–44, 1992.

[128] D. Kunz. Channel assignment for cellular radio using neural networks. *IEEE Transactions on Vehicular Technology*, 40:188–193, 1991.

[129] W.K. Lai and G.G. Coghill. Channel assignment through evolutionary optimization. *IEEE Transactions on Vehicular Technology*, 45:91–95, 1996.

[130] T.A. Lanfear. Graph theory and radio frequency assignment. Technical Report NATO Unclassified, Allied Radio Frequency Agency (ARFA), 1989.

[131] R.L.M.J. van de Leensel. *Models and Algorithms for Telecommunication Network Design*. PhD thesis, Maastricht University, 1999.

[132] R.L.M.J. van de Leensel, O.E. Flippo, and A.M.C.A. Koster. A dynamic programming algorithm for the ATM network installation problem on a tree. Research Memorandum 98/009, Maastricht University, 1998.

[133] D.D.-F. Liu. T-graphs and the channel assignment problem. *Discrete Mathematics*, 161:197–205, 1996.

[134] G.D. Lochtie and M.J. Mehler. Channel assignment using a subspace approach to neural networks. *IEE Conference Publication*, 407:296–300, 1995.

[135] G.D. Lochtie and M.J. Mehler. Subspace approach to channel assignment in mobile communication networks. *IEE Proceedings*, 142:179–185, 1995.

[136] T.L. Magnanti and R.T. Wong. Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research*, 29:464–484, 1981.

[137] E. Malesińska. List coloring and optimization criteria for a channel assignment problem. Technical Report 458, Technische Universität Berlin, 1995.

[138] E. Malesińska. *Graph-Theoretical Models for Frequency Assignment Problems*. PhD thesis, Technischen Universität Berlin, 1997.

[139] E. Malesińska and A. Panconesi. On the hardness of allocating frequencies for hybrid networks. Technical Report 498, Technische Universität Berlin, 1996.

[140] C. Mannino and A. Sassano. An enumerative algorithm for the frequency assignment problem. Technical Report 1096, Università di Roma La Sapienza, 1998.

[141] R. Mathar and J. Mattfeldt. Channel assignment in cellular radio networks. *IEEE Transactions on Vehicular Technology*, 42:647–656, 1993.

[142] C. McDiarmid. A doubly cyclic channel assignment problem. *Discrete Applied Mathematics*, 80:263–268, 1998.

[143] A. Mehrotra and M.A. Trick. A column generation approach for graph coloring. *INFORMS Journal on Computing*, 8:344–354, 1996.

[144] K. Menger. Zur allgemeinen kurventheorie. *Fundamenta Mathematicae*, 10:96–115, 1927.

[145] B.H. Metzger. Spectrum management technique, Fall 1970. Presentation at 38th National ORSA meeting (Detroit, MI).

[146] R. Müller and A.S. Schulz. Transitive packing. *Lecture notes in computer science*, 1084:430–444, 1996.

[147] R.A. Murphey, P.M. Pardalos, and M.G.C. Resende. Frequency assignment problems. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of combinatorial optimization*, volume 3. Kluwer Academic Publishers, 1999. To appear.

[148] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization*. Wiley, N.Y., 1988.

[149] Y. Nesterov and A. Nemirovskii. *Interior-Point Polynomial Algorithms in Convex Programming*. SIAM studies in applied mathematics, Philadelphia, 1994.

[150] C. Y. Ngo and V.O.K. Li. Fixed channel assignment in cellular radio networks using a modified genetic algorithm. *IEEE Transactions on Vehicular Technology*, 47:163–171, 1998.

[151] Jaarverslag 1998 Onafhankelijke Post en Telecommunicatie Autoriteit (OPTA), 1998. Annual report Netherlands Regulatory Authority for the Telecommunications and Postal sector, available at http://www.opta.nl.

[152] M. Padberg. The boolean quadric polytope: Some characteristics, facets and relatives. *Mathematical Programming*, 45:139–172, 1989.

[153] C.H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization : Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, 1985.

[154] T. Park and C.Y. Lee. Application of the graph coloring algorithm to the frequency assignment problem. *Journal of the Operations Research Society of Japan*, 39:258–265, 1996.

[155] D.V. Pasechnik. An interior point approximation algorithm for a class of combinatorial optimization problems: Implementation and enhancements. Technical report, Delft University of Technology, 1998.

[156] J. Picard and D. Ratliff. Minimum cuts and related problems. *Networks*, 5:357–370, 1975.

[157] A. Quellmalz, A. Knälmann, and B. Müller. Efficient frequency assignment with simulated annealing. *IEE conference publication*, 407-2:301–304, 1995.

[158] A. Raychaudhuri. *Intersection Assignments, T-Colourings and Powers of Graphs.* PhD thesis, Rutgers University, 1985.

[159] J. Rhys. A selection problem of shared fixed costs and networks. *Management Science*, 7:200–207, 1970.

[160] F.S. Roberts. T-colorings of graphs: Recent results and open problems. *Discrete Mathematics*, 93:229–245, 1991.

[161] N. Robertson and P.D. Seymour. Graph minors. I. excluding a forest. *Journal of Combinatorial Theory, Series B*, 35:39–61, 1983.

[162] N. Robertson and P.D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *Journal of Algorithms*, 7:309–322, 1986.

[163] A.N. Rouskas, M.G. Kazantzakis, and M.E. Anagnostou. Optimal channel assignment in cellular networks. *International Journal of Communication Systems*, 8:359–364, 1995.

[164] J.H.G.C. Rutten. *Polyhedral clustering.* PhD thesis, Universiteit Maastricht, 1998.

[165] H.G. Sandalidis, P.P. Stavroulakis, and J. Rodriguez-Tellez. Borrowing channel assignment strategies based on heuristic techniques for cellular systems. *IEEE Transactions on Neural Networks*, 10:176–181, 1999.

[166] A. Schrijver. *Theory of linear and integer programming.* Wiley, New York, 1986.

[167] M.A. Shepherd. *Radio Channel Assignment.* PhD thesis, Oxford Univerisity, 1998.

[168] H.D. Sherali, Y. Lee, and W.P. Adams. A simultaneous lifting strategy for identifying new classes of facets for the boolean quadric polytope. *Operations Research Letters*, 17:19–26, 1995.

[169] C. De Simone. The cut polytope and the boolean quadric polytope. *Discrete Mathematics*, 79:71–75, 1989.

[170] C. De Simone. A note on the boolean quadric polytope. *Operations Research Letters*, 19:115–116, 1996.

[171] K.N. Sivarajan, R.J. McEliece, and J.W. Ketchum. Channel assignment in cellular radio. In *Proceedings of the 39th IEEE Vehicular Technology Conference*, pages 846–850, 1989.

[172] D.H. Smith and S. Hurley. Bounds for the frequency assignment problem. *Discrete Mathematics*, 167/168:571–582, 1997.

[173] D.H. Smith, S. Hurley, and S.U. Thiel. Improving heuristics for the frequency assignment problem. *European Journal of Operational Research*, 107:76–86, 1998.

[174] K.A. Smith. A genetic algorithm for the channel assignment problem. *IEEE Global Communications Conference*, 4:2013–2017, 1998.

[175] K.A. Smith, B.K. Kim, and G.F. Sargent. Minimising channel interference in real cellular radio networks. *IEEE Global Communications Conference*, 4:2192–2197, 1998.

[176] K.A. Smith and M. Palaniswami. Static and dynamic channel assignment using neural networks. *IEEE Journal on Selected Areas in Communications*, 15:238–249, 1997.

[177] J.F. Sturm. Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones. Technical report, Communications Research Laboratory, McMaster University, Hamilton, Canada, 1998. Available at: http://www.unimaas.nl/~sturm/.

[178] J.F. Sturm. Private communication, 1999.

[179] C.W. Sung and W.S. Wong. Sequential packing algorithm for channel assignment under cochannel and adjacent channel interference constraint. *IEEE Transactions on Vehicular Technology*, 46:676–685, 1997.

[180] D. Tcha, Y. Chung, and T. Choi. A new lower bound for the frequency assignment problem. *IEEE/ACM Transactions on Networking*, 5:34–39, 1997.

[181] H. Thuve. Frequency planning as a set partitioning problem. *European Journal of Operational Research*, 6:29–37, 1981.

[182] S.R. Tiourine. *Decision Support by Combinatorial Optimization: Case Studies.* PhD thesis, Eindhoven University of Technology, 1999.

[183] S.R. Tiourine, C.A.J. Hurkens, and J.K. Lenstra. An overview of algorithmic approaches to frequency assignment problems. In *Calma Symposium on Combinatorial Algorithms for Military Applications*, pages 53–62, 1995. Available at http://www.win.tue.nl/math/bs/ comb_opt/hurkens/calma.html.

[184] S.R. Tiourine, C.A.J. Hurkens, and J.K. Lenstra. Local search algorithms for the radio link frequency assignment problem. *Telecommunication systems*, to appear, 1999.

[185] E.P.K. Tsang. *Foundations of Constraint Satisfaction.* Academic Press, London, 1993.

[186] C. Valenzuela, S. Hurley, and D.H. Smith. A permutation based genetic algorithm for minimum span frequency assignment. *Lecture Notes in Computer Science*, 1498:907–916, 1998.

[187] G. Verfaillie, M. Lemaître, and T. Schiex. Russian doll search for solving constraint optimization problems. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96)*, pages 181–187, Portland, OR, USA, 1996.

[188] V.G. Vizing. Critical graphs with given chromatic class (russian). *Diskret. Analiz*, 5:9–17, 1965.

[189] R.J. Wallace and E.C. Freuder. Heuristic methods for over-constrained constraint satisfaction problems. *Lecture Notes in Computer Science*, 1106:207–216, 1996.

[190] J.P. Walser. Feasible cellular frequency assignment using constraint programming abstractions. In *Proceedings of the Workshop on Constraint Programming Applications (CP96)*, Cambridge, Massachusetts, USA, 1996.

[191] W. Wang and C.K. Rushforth. An adaptive local-search algorithm for the channel-assignment problem (CAP). *IEEE Transactions on Vehicular Technology*, 45:459–466, 1996.

[192] J.P. Warners. A potential reduction approach to the radio link frequency assignment problem. Master's thesis, Delft University of Technology, 1995.

[193] J.P. Warners, T. Terlaky, C. Roos, and B. Jansen. Potential reduction algorithms for structured combinatorial optimization problems. *Operations Research Letters*, 21:55–64, 1997.

[194] J.P. Warners, T. Terlaky, C. Roos, and B. Jansen. A potential reduction approach to the frequency assignment problem. *Discrete Applied Mathematics*, 78:251–282, 1997.

[195] D. De Werra and Y. Gay. Chromatic scheduling and frequency assignment. *Discrete Applied Mathematics*, 49:165–174, 1994.

[196] D. Youngs. Frequency assignment for cellular radio networks. *IEE Conference Publication*, 404:179–183, 1995.

[197] J.A. Zoellner and C.L. Beall. A breakthrough in spectrum conserving frequency assignment technology. *IEEE Transactions on Electromagnetic Compatiblity*, 19:313–319, 1977.

Author index

Auteur index

Aardal, K.I. 11, 20, 22, 23, 24, 31, 32, 40, 41, 42, 49, 105
Aarts, E.H.L. 11
Adams, W.P. 76
Ahuja, R.K. 12, 87
Al-Khaled, F.S. 45
Allen, S.M. 13, 31, 46
Anagnostou, M.E. 37
Anderson, L.G. 18, 28
Arnborg, S. 86

Balinski, M. 67
Barahona, F. 67
Bater, J. 46
Beall, C.L. 32
Benders, J.F. 125
Benthem, H.P. van 19
Bienstock, D. 64
Biró, M. 22
Bodlaender, H.L. 83, 85, 86
Borndörfer, R. 19, 44, 129, 131
Bouju, A. 23, 24, 31, 33
Box, F. 32
Boyce, J.F. 23, 24, 31, 33
Brélaz, D. 26, 44
Brockmüller, B. 64

Cabon, B. 20
Castelino, D.J. 18, 43, 45
Chang, K.-N. 35, 36, 37
Chardaire, P. 24, 25, 41, 42
Choi, T. 31, 32
Chung, Y. 31, 32
Coghill, G.G. 43
Cohen, D. 46
Cook, W. 8
Corneil, D.G. 86
Costa, D. 44
Cozzens, M.B. 21
Crisan, C. 19, 24, 25, 43
Cuppini, M. 25

Dahlhaus, E. 69
Dayhoff, J.E. 12
Demestichas, P.P. 37
Dimitropoulos, C.H.D. 23, 24, 31, 33
Dorne, R. 19, 43
Dowd, P. 43
Dunkin, N. 13, 46
Duque-Antón, M. 45

Eisenblätter, A. 19, 44, 129, 131
Erdös, P. 21

Fischetti, M. 13, 36, 46
Fisher, M.L. 126
Flippo, O.E. ii
Freuder, E.C. 50
Funabiki, N. 44, 46

Galinier, P. 19, 43
Gamst, A. 29, 30, 32, 43
Garey, M.R. 11, 21, 52, 67
Gay, Y. 26
Gelatt, C.D. 11
Geoffrion, A.M. 127
Giortzis, A.I. 32, 37
Givry, S. De 20, 41, 42
Glover, F. 11, 43
Goldberg, D.E. 12
Gomory, R.E. 88
Griggs, J.R. 26
Grötschel, M. 19, 44, 129, 131
Günlük, O. 64

Hale, W.K. 17, 21, 26
Hansen, P. 67

Hao, J.-K. 19, 43
Held, M. 126
Helmberg, C. 127
Heuvel, J. van den 46
Hipolito, A. 22, 23, 24, 31, 32, 40, 41, 42, 105
Hoesel, C.P.M. van ii, 11, 22, 23, 24, 31, 32, 40, 41, 42, 49, 84, 105, 107
Holland, J.H. 12
Hopfield, J.J. 12
Hu, T.C. 69, 88
Hujter, M. 22
Hurkens, C.A.J. 20, 23, 24, 25, 31, 32, 33, 41, 42, 105
Hurley, S. 18, 28, 30, 31, 32, 43, 45

Jansen, B. 22, 23, 24, 31, 32, 40, 41, 42, 105
Janssen, J. 14, 30, 31
Jaumard, B. 11, 37
Jeavons, P. 46
Johnson, D.S. 11, 21, 52, 67, 69
Johri, P.K. 14

Kamath, A.P. 25
Kapsalis, A. 24, 25, 41, 42
Karmarkar, N.K. 25
Karp, R.M. 21, 126
Katzela, I. 15
Kazantzakis, M.G. 37
Ketchum, J.W. 30, 32, 43
Kilakos, K. 14, 30, 31
Kim, B.K. 44
Kim, J.-S. 43
Kim, S. 33, 35, 36, 37
Kim, S.-L. 33
Kirkpatrick, S. 11
Klabjan, D. 63
Knälmann, A. 45
Kolen, A.W.J. ii, 23, 24, 31, 32, 41, 42, 45, 49, 77, 84, 105, 107, 109, 118, 120, 121, 123, 132, 158
Korst, J. 11
Koster, A.M.C.A. ii, 11, 41, 49, 84, 107
Kubale, M. 26
Kumar, V. 50
Kunz, D. 44, 45

Laguna, M. 11
Lai, W.K. 43
Lanfear, T.A. 33
Lee, C.Y. 44
Lee, Y. 76
Leensel, R.L.M.J. van de ii
Leese, R.A. 46
Lemaître, M. 41
Lenstra, J.K. 20, 23, 24, 25, 31, 33, 41, 42, 105
Lepschy, C. 13, 36, 46
Li, V.O.K. 43
Likas, A. 23, 24, 31, 33
Liu, D.D.-F. 26
Lobjois, L. 20
Lochtie, G.D. 44

Magnanti, T.L. 12, 87, 126
Malesińska, E. 46
Mannino, C. 11, 20, 36, 37
Marcotte, O. 11, 14, 37
Martin, A. 19, 44, 129, 131
Mathar, R. 36, 37
Mattfeldt, J. 36, 37
McDiarmid, C. 46
McEliece, R.J. 30, 32, 43
Mehler, M.J. 44
Mehrotra, A. 37
Menger, K. 87
Metzger, B.H. 17, 21
Meyer, C. 11, 37
Minerva, G. 13, 36, 46
Mühlenbein, H. 19, 24, 25, 43
Müller, B. 45
Müller, R. 52
Murphey, R.A. 26

Naghshineh, M. 15
Nasrabadi, N. 43
Nemhauser, G.L. 12, 49, 57, 63
Nemirovskii, A. 127, 128
Nesterov, Y. 127, 128
Ngo, C. Y. 43
Nishikawa, S. 46

Orlin, J.B. 12, 87

Padberg, M. 52, 74, 75, 76
Palaniswami, M. 44
Panconesi, A. 46
Papadimitriou, C.H. 11, 69
Papageorgiou, G. 23, 24, 31, 33
Pardalos, P.M. 26
Park, S. 43
Park, T. 44
Pasechnik, D.V. 24, 25, 31, 33, 41, 42
Picard, J. 67
Proskurowski, A. 86

Quellmalz, A. 45

Ramakrishnan, K.G. 25
Ratliff, D. 67
Raychaudhuri, A. 31
Rayward-Smith, V.J. 24, 25, 41, 42
Resende, M.G.C. 25, 26
Rhys, J. 67
Roberts, F.S. 21, 26, 31
Robertson, N. 83, 84, 85
Rodriguez-Tellez, J. 14
Romanin Jacur, G. 13, 36, 46
Roos, C. 23, 24, 31, 41, 42
Rouskas, A.N. 37
Rüber, B. 45
Rubin, A.L. 21
Rushforth, C.K. 30, 33
Rutten, J.H.G.C. 64

Sandalidis, H.G. 14
Sargent, G.F. 44
Sassano, A. 11, 20, 36, 37
Savelsbergh, M.W.P. ii
Scheidt, G. Vom 23, 24, 31, 33
Schiex, T. 20, 41, 42
Schrijver, A. 12, 49
Schulz, A.S. 52
Seymour, P.D. 69, 83, 84, 85
Shepherd, M.A. 46
Sherali, H.D. 76
Simone, C. De 76
Sivarajan, K.N. 30, 32, 43
Smith, D.H. 28, 30, 31, 32
Smith, G.D. 24, 25, 41, 42
Smith, K.A. 43, 44
Stafylopatis, A. 23, 24, 31, 33
Stavroulakis, P.P. 14
Steiglitz, K. 11
Stephens, N.M. 18, 43, 45
Stockmeyer, L. 52
Sturm, J.F. 129
Sung, C.W. 30, 31, 32, 33

Takefuji, Y. 44
Tank, D.W. 12
Taylor, H. 21
Taylor, J.G. 23, 24, 31, 33
Tcha, D. 31, 32
Terlaky, T. 23, 24, 31, 41, 42
Thiel, S.U. 28, 30, 31, 32
Thuve, H. 20, 46
Tiourine, S.R. 20, 23, 24, 25, 31, 32, 33, 41, 42, 105
Toto, E. 13, 36, 46
Tovey, C. 63
Trick, M.A. 37
Tsang, E.P.K. 50
Turner, L.F. 32, 37
Tuza, Zs. 22

Valenzuela, C. 28, 30, 32
Vecchi, M.D. 11
Verfaillie, G. 41, 42
Vizing, V.G. 21
Vovor, T. 37

Wal, R. van der 23, 24, 31, 32
Wallace, R.J. 50
Walser, J.P. 45
Wang, W. 30, 33
Warners, J.P. 20, 23, 24, 31, 33, 41, 42
Werra, D. De 26
Wolsey, L.A. 12, 49, 57, 64
Wong, R.T. 126
Wong, W.S. 30, 31, 32, 33

Yannakakis, M. 69
Youngs, D. 44

Zoellner, J.A. 32

Samenvatting in het Nederlands

Summary in Dutch

Introductie

In 1909 ontving de Italiaan Marconi de Nobelprijs voor zijn baanbrekende werk op het gebied van de draadloze telegraaf. Sinds het eind van de 19e eeuw experimenteerde hij met het verzenden van boodschappen door middel van radiogolven. Sinds die tijd heeft de draadloze communicatie een grote vlucht genomen. Reeds in 1915 was het mogelijk om gesproken woord over de Atlantische oceaan te verzenden. Niet lang daarna werd radio gemeengoed en al halverwege de jaren '30 werd volop geëxperimenteerd met het verzorgen van televisie uitzendingen. Na de tweede wereldoorlog was de technologie zo ver gevorderd dat televisie op grote schaal kon worden geïntroduceerd.

Direct na de tweede wereldoorlog startte ook de exploitatie van het eerste draadloze telefoonnetwerk in de Verenigde Staten. Het duurde tot het begin van de jaren '80 voordat de mobiele telefoon op grote schaal zijn intrede deed. Nadeel van de eerste mobiele netwerken was de diversiteit aan technologieën die werden gebruikt. Daardoor was het in Europa onmogelijk de mobiele telefoon in het buitenland te gebruiken. Standaardisatie van de technologie werd dan ook de opdracht voor de in 1988 opgerichte *Groupe Speciale Mobile* (GSM). In 1992 werd het eerste GSM-netwerk geïntroduceerd in Duitsland. Inmiddels is GSM een groot succes gebleken en beschikken wereldwijd tientallen miljoenen mensen over een aansluiting op een GSM-netwerk (zie Figuur 1.2, pagina 4). Naast radio, televisie en mobiele telefonie, heeft draadloze communicatie ook toepassingen in de ruimtevaart, luchtverkeersbegeleiding en militaire communicatie systemen.

Draadloze communicatie vindt plaats met behulp van een radiozender (transmitter) en ontvanger (receiver). De transmitter moduleert een frequentie uit het radiospectrum. De receiver zet de modulatie van de frequentie om naar geluid en/of beeld. Wanneer twee communicatieverbindingen in dezelfde regio gebruik maken van (bijna) dezelfde frequentie kan interferentie van het signaal optreden. D.w.z. dat de kwaliteit van het door de receiver ontvangen signaal verslechtert. Afhankelijk van het niveau van de interferentie kan de kwaliteit van het signaal als onacceptabel worden gekwalificeerd. In de praktijk betekent dit dat voor de twee communicatieverbindingen frequenties moeten worden gebruikt die minimaal een bepaalde afstand tot elkaar hebben. Echter, door de vele toepassingsgebieden van draadloze communicatie en de begrensdheid van het radiospectrum zijn slechts een beperkt aantal frequenties beschikbaar voor elke vorm van draadloze communicatie.

Dit betekent dat hergebruik van frequenties binnen één en dezelfde geografische regio noodzakelijk is. Het hergebruik van frequenties kan echter leiden tot de reeds genoemde interferentie. De toewijzing van frequenties aan communicatieverbindingen zal dan ook zorgvuldig moeten gebeuren om de interferentie tot een minimum te beperken. De afweging tussen hergebruik van frequenties en de kwaliteit van het telecommunicatienetwerk staat bekend als het frequentie toewijzingsprobleem. Kwantificatie van de diverse aspecten van het frequentie toewijzingsprobleem leidt tot een wiskundig optimaliseringsprobleem dat met behulp van technieken uit de besliskunde kan worden opgelost.

Frequentie Toewijzing: Een Overzicht

Door de sterke groei van mobiele telecommunicatie in het laatste decennium is de aandacht voor het frequentie toewijzingsprobleem de laatste jaren sterk toegenomen. Afhankelijk van de specifieke eigenschappen van een netwerk en de doelstelling van het onderzoek zijn veel verschillende wiskundige modellen en besliskundige oplossingstechnieken voorgesteld. Na de introductie in **Hoofdstuk 1**, geeft **Hoofdstuk 2** een overzicht van de literatuur die de laatste jaren is verschenen op het gebied van frequentie toewijzing. Hierbij kunnen wij onderscheid maken tussen vaste, dynamische en hybride toewijzingsschema. In Hoofdstuk 2 beperken wij ons tot modellen en methoden voor vaste toewijzingsschema's. Het frequentie toewijzingsprobleem kan als een wiskundig optimalisatieprobleem worden geformuleerd door middel van een graaf. Elke knoop in de graaf correspondeert met een draadloze verbinding, terwijl een kant in de graaf aangeeft dat de aanliggende verbindingen kunnen interfereren afhankelijk van de keuze van de frequenties. Wanneer meerdere verbindingen tussen dezelfde geografische locaties moeten worden gerealiseerd wordt veelal slechts één knoop voor alle verbindingen gemodelleerd. Aan deze knoop moeten dan meerdere frequenties worden toegewezen. Voor elke knoop is een eindige set van frequenties beschikbaar, welke kan verschillen van knoop tot knoop. Het kan bijvoorbeeld zo zijn dat bepaalde frequenties in de omgeving van landsgrenzen niet mogen worden gebruikt vanwege bilaterale afspraken.

Afhankelijk van de doelstellingsfunctie kunnen de modellen voor het frequentie toewijzingsprobleem in 4 categorieën worden geclassificeerd:

- minimalisatie van het aantal gebruikte frequenties in een interferentievrije toewijzing (Minimum Order Frequency Assignment Problem, MO-FAP),
- minimalisatie van het gebruikte frequentie interval in een interferentievrije toewijzing (Minimum Span Frequency Assignment Problem, MS-FAP),
- minimalisatie van de totale blokkeringskans van het netwerk bij een interferentievrije toewijzing (Minimum Blocking Frequency Assignment Problem, MB-FAP), en

- minimalisatie van de totale interferentie in een toewijzing (Minimum Interference Frequency Assignment Problem, MI-FAP).

Het graaf kleuringsprobleem kan wiskundig als een speciaal geval van frequentie toewijzing worden gezien. Hieruit volgt dat elk van de bovenstaande problemen $\mathcal{NP}$-moeilijk is, hetgeen wil zeggen dat het onwaarschijnlijk is dat er optimale algoritmen bestaan met rekentijd cq. geheugengebruik polynomiaal in de gegevens. Voor elk van de 4 modellen wordt in Hoofdstuk 2 een probleemdefinitie, een wiskundige formulering en een overzicht van de toegepaste technieken gegeven. Voor zover mogelijk worden de verschillende methoden met elkaar vergeleken op dezelfde probleeminstanties. Voor het MO-FAP, MS-FAP en MI-FAP zijn instanties van het CALMA-project vrij beschikbaar [32]. Daarnaast zijn voor het MS-FAP de zogenaamde Philadelphia-instanties beschikbaar (zie Figuur 2.1, pagina 28). Tot voor kort was het gebruikelijk om elke antenne in een mobiel telefoonnetwerk te representeren door middel van een hexagoon. Het aantal frequenties dat moet worden toegewezen, in zowel de CALMA- en Philadelphia-instanties als andere praktijkinstanties, varieert tussen honderd en enige duizenden. Het beschikbare aantal frequenties loopt uiteen van een tiental tot enkele honderden (Philadelphia-instanties).

In het MO-FAP moeten de frequenties worden toegewezen zodanig dat

(i). alle verbindingen kunnen worden gerealiseerd,

(ii). er geen interferentie ontstaat, en

(iii). het aantal gebruikte frequenties minimaal is.

De meeste moderne heuristische methoden uit de besliskunde en kunstmatige intelligentie zijn toegepast op het MO-FAP. Meerdere studies tonen aan dat met tabu search de optimale oplossing regelmatig te genereren. Ook de minder bekende potentiaal reductiemethode gebaseerd op inwendige puntmethoden lijkt goede resultaten op te leveren. Exacte methoden, zoals geheeltallig lineair programmeren en constraint satisfaction, blijken voor de meeste van de beschikbare instanties krachtig genoeg een oplossing te genereren waarvoor optimaliteit kan worden bewezen. Ook door combinatie van ondergrenzen, gebaseerd op klieks in de graaf, met een heuristiek zoals tabu search, kan in veel gevallen optimaliteit van de gegenereerde oplossing worden bewezen.

In het MS-FAP moeten de frequenties worden toegewezen zodanig dat

(i). alle verbindingen kunnen worden gerealiseerd,

(ii). er geen interferentie ontstaat, en

(iii). het verschil tussen het maximum en minimum van de gebruikte frequenties minimaal is.

Voor het MS-FAP heeft het meeste onderzoek zich gericht op ondergrenzen voor de Philadelphia-instanties. De ondergrenzen zijn ofwel gebaseerd op graaf theoretische argumenten ofwel op oplossingen/ondergrenzen voor een gerelateerd handelsreizigersprobleem. Met name de laatste grenzen (veelal toegepast op een deelgraaf) zijn vaak krachtig genoeg om gecombineerd met heuristieken optimaliteit te bewijzen. De instanties van het CALMA-project zijn helaas niet moeilijk genoeg om onderscheid te maken tussen de kwaliteit van de verschillende technieken.

In het MB-FAP moeten de frequenties worden toegewezen zodanig dat

(i). er geen interferentie ontstaat, en

(ii). de kans op blokkering van een verbinding minimaal is.

Het MB-FAP kan worden beschouwd als een generalisatie van het *independent set* probleem, één van de standaardproblemen in de combinatorische optimalisering. Vandaar dat het onderzoek voor het MB-FAP zich met name heeft gericht op exacte methoden gebaseerd op geheeltallige formuleringen. In de meeste gevallen zijn deze exacte methoden krachtig genoeg om de instanties tot optimaliteit op te lossen. Heuristische methoden zijn slechts op beperkte schaal toegepast op deze variant van frequentie toewijzing.

In het MI-FAP moeten de frequenties zodanig worden toegewezen dat

(i). alle verbindingen kunnen worden gerealiseerd, en

(ii). de totale interferentie minimaal is.

Het MI-FAP blijkt niet alleen het meest algemene model te zijn, maar ook het moeilijkst oplosbare. Vandaar dat het onderzoek voor dit probleem zich vooral op heuristieken heeft gericht. Met name tabu search en genetische algoritmen zijn veelvuldig toegepast op verschillende praktijkinstanties. Voor de beschikbare CALMA-instanties zijn de beste resultaten behaald met een speciaal genetisch algoritme waarin oplossingen optimaal worden gekruist. Hierbij is gebruik gemaakt van de resultaten van hoofdstuk 3 van dit proefschrift. Voor slechts een tweetal speciale instanties zijn ondergrenzen beschikbaar. Voor de overige instanties ontbreken ondergrenzen in het geheel of zijn zeer zwak. Voor de kleinste instantie is de optimale oplossing bekend na toepassing van een zeer rekenintensieve constraint satisfaction techniek. Voor andere praktijkinstanties zijn geen ondergrenzen beschikbaar.

Exacte Methoden voor het MI-FAP

Het gebrek aan goede ondergrenzen voor het minimum interferentie frequentie toewijzingsprobleem, is de aanleiding geweest voor het onderzoek dat gepresenteerd wordt in de Hoofdstukken 3 en 4. In **Hoofdstuk 3** wordt het MI-FAP gemodelleerd als *Partial Constraint Satisfaction Problem* (PCSP). Wij bestuderen het PCSP vanuit een polyhedraal oogpunt. Allereerst wordt het PCSP geformuleerd als een geheeltallig lineair programmeringsprobleem. Na bepaling van de dimensie van het bijbehorende polytoop en beschrijving van de ongelijkheden die triviale facetten beschrijven, vervolgen we de discussie met twee stellingen voor het liften van toegestane ongelijkheden. Deze stellingen bieden de mogelijkheid om klassen van facet definiërende ongelijkheden af te leiden uit individuele ongelijkheden. Op deze wijze worden 2 klassen van ongelijkheden afgeleid. Voor deze ongelijkheden bespreken wij de complexiteit van de bijbehorende separatieproblemen. Nieuwe klassen van ongelijkheden kunnen ook worden afgeleid door de overeenkomsten tussen het PCSP en het *Boolean Quadric Polytope* te beschouwen. Hoofdstuk 3 wordt afgesloten met rekenresultaten die het nut van de geldende ongelijkheden aantonen. Voor PCSP's met domeinen van beperkte grootte kan de optimale oplossing met behulp van een *cutting plane* algoritme in acceptabele tijd worden verkregen. Voor PCSPs met grotere domeinen is deze methode helaas niet krachtig genoeg om in een redelijke tijd goede resultaten op te leveren.

In **Hoofdstuk 4** wordt daarom een andere methode toegepast waarin de onderliggende graafstructuur van het probleem beter wordt benut. De methode is gebaseerd op de boombreedte van de graaf. Voor vele combinatorische optimaliseringsproblemen gebaseerd op een graaf bestaan optimale algoritmen die polynomiaal zijn in tijd en geheugen, zolang de boombreedte van de graaf beperkt is tot een constante. Ook voor het MI-FAP bestaat een dynamisch programmeringsalgoritme dat gebruik maakt van een boomdecompositie met beperkte breedte. Voor het bepalen van een boomdecompositie van beperkte breedte bestaat in principe een polynomiaal algoritme. Aangezien het algoritme in de praktijk van weinig waarde is (exponentieel in de breedte), beschrijven we in Hoofdstuk 4 allereerst een heuristiek voor het bepalen van een boomdecompositie. Vervolgens presenteren we een dynamisch programmeringsalgoritme voor het MI-FAP. Zoals alle algoritmen gebaseerd op een beperkte boombreedte is ook dit algoritme echter exponentieel in de breedte van de boomdecompositie. Hierdoor kunnen praktijkinstanties alleen worden opgelost met behulp van extra reductietechnieken. We beschrijven zowel graaf-reductietechnieken als reductietechnieken die het aantal domeinelementen verkleinen. Deze laatste techniek kan ook gedurende het dynamisch programmeringsalgoritme worden toegepast op toewijzingen aan een deelverzameling van de knopen. De graaf-reductietechnieken worden met name in de preprocesfase gebruikt. Met behulp van deze reductietechnieken en het dynamisch programmeringsalgoritme kunnen 7 van de 11 CALMA-instanties worden opgelost.

Voor de overige 4 instanties blijkt het algoritme echter nog steeds vast te lopen op een

tekort aan geheugen en rekentijd. Vandaar dat een iteratieve versie van het boomdecompositie algoritme wordt gepresenteerd. Dit algoritme genereert een niet-dalende reeks van ondergrenzen. Allereerst wordt het domein voor elke knoop in de graaf verdeeld in een klein aantal deelverzamelingen. Vervolgens wordt een PCSP opgelost waarin een keuze tussen deze deelverzamelingen moet worden gemaakt. De kostencoëfficiënten voor dit PCSP worden zodanig gekozen dat de optimale oplossing een ondergrens voor het oorspronkelijke probleem oplevert. Verkleining van de deelverzamelingen levert ondergrenzen op die niet slechter zijn dan de eerste. Op deze wijze worden voor de overige 4 instanties redelijk tot zeer goede ondergrenzen verkregen. Tot slot combineren we het iteratieve algoritme met de geheeltallige programmeringstechnieken van Hoofdstuk 3, hetgeen in enkele gevallen nog betere ondergrenzen oplevert.

Heuristieken voor MI-FAP

In **Hoofdstuk 5** wordt het nut van de technieken uit de Hoofdstukken 3 en 4 voor heuristische methoden besproken. Vaak kunnen exacte methoden ook worden gebruikt in een heuristisch kader. In Kolen [118] zijn de polyhedrale resultaten van Hoofdstuk 3 reeds gebruikt om een optimale kruising van oplossingen in een genetisch algoritme mogelijk te maken. Op deze wijze werden de best bekende oplossingen voor alle CALMA-instanties verkregen (in veel gevallen inmiddels bewezen optimaal door de resultaten van Hoofdstuk 4). In Hoofdstuk 5 worden de polyhedrale resultaten gecombineerd met een lokaal zoekalgoritme. Wij definiëren de buurruimte van een oplossing zodanig dat de beste buur verkregen kan worden door het oplossen van een PCSP met 2 frequenties per domein. De resultaten van één van de geteste varianten benaderen in kwaliteit de resultaten van het genetisch algoritme.

De resultaten van Hoofdstuk 4 kunnen eveneens met een lokaal zoekalgoritme worden gecombineerd. In plaats van een buurruimte waarin slechts één frequentie per keer kan worden veranderd, worden buurruimtestructuren voorgesteld waarin de toewijzing van een deelgraaf kan worden gewijzigd. Voorwaarde is dat de deelgraaf een beperkte boombreedte heeft. Voorlopige rekenresultaten tonen aan dat de lokaal optimale oplossingen met deze uitgebreidere buurruimtestructuren duidelijk betere oplossingen opleveren.

Richtingen voor Nader Onderzoek en Conclusies

Dit proefschrift wordt afgesloten met een aantal suggesties voor richtingen waarin nader onderzoek op het MI-FAP mogelijk is. Mogelijkheden om Benders Decompositie, Lagrange Relaxatie en een Semi Definite Programmering Relaxatie toe te passen worden

kort besproken. Ook wordt een nieuwe geheeltallige programmering formulering gepresenteerd die aanzienlijk minder variabelen en restricties heeft dan de formulering van Hoofdstuk 3. Met name de combinatie van boomdecompositie met Langrange Relaxatie, en de nieuwe formulering mogen worden beschouwd als waardevolle richtingen voor nader onderzoek. Tot slot worden de resultaten van dit proefschrift samengevat (zie Tabel 6.2).

Curriculum Vitae

Levensloop

Arie Koster was born on August 10, 1973 in Gouda, and grew up in Schoonhoven, the Netherlands. After completion of the VWO (pre-university education) at the Christelijke Scholengemeenschap "Willem de Zwijger" in Schoonhoven, he started in 1991 his studies in Technical Mathematics at Delft University of Technology. He specialized in Operations Research, and graduated in 1995 cum laude at the department of Statistics, Probability Theory and Operations Research on *DualNet*, a software package for the graphical representation of minimum cost flow network problems and algorithms. From September 1995, he is appointed as a Ph.D. student at the department of Quantitative Economics of the Universiteit Maastricht, where he completed his Doctoral thesis. The research presented in the thesis is part of the project "Combinatorial Optimization problems in the Telecommunication". From December 1, 1999, Arie Koster will be affiliated for 2 years as postdoctoral researcher on Operations Research and Telecommunication at the Konrad Zuse Zentrum für Informationstechnik Berlin (ZIB).

Arie Koster werd geboren op 10 augustus 1973 in Gouda, en groeide op in Schoonhoven. Na voltooiing van het VWO aan de Christelijke Scholengemeenschap "Willem de Zwijger" in Schoonhoven, begon hij in 1991 de studie Technische Wiskunde aan de Technische Universiteit Delft. Hij specialiseerde zich in de Besliskunde, en studeerde in 1995 cum laude af bij de vakgroep Statistiek, Stochastiek en Operationele Analyse op *DualNet*, een software pakket voor het visualiseren van minimale kosten netwerk stroomproblemen en algoritmen. Sinds September 1995 is hij als assistent in opleiding aangesteld bij de vakgroep Kwantitatieve Economie van de Universiteit Maastricht, alwaar hij zijn proefschrift voltooide. Het onderzoek gepresenteerd in het proefschrift is onderdeel van het project "Combinatorische Optimalisering problemen in de Telecommunicatie". Per 1 December 1999 zal Arie Koster voor 2 jaar verbonden zijn aan het Konrad Zuse Zentrum für Informationstechnik Berlin (ZIB) als postdoctoraal onderzoeker op het gebied van besliskunde en telecommunicatie.